University of Alberta

Library Release Form

Name of Author : Laura Choy Lai Fun

Title of Thesis: Turbo Codes for Advanced Wireless Access Systems

Degree: Master of Science

Year this Degree Granted: 2002

University of Alberta

Turbo Codes for Advanced Wireless Access Systems

by

Laura Choy Lai Fun

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of Master of Science

Department of Electrical and Computer Engineering,

Edmonton, Alberta

Spring 2002.

University of Alberta

Faculty of Graduate Studies and Research

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled "Turbo Codes for Advanced Wireless Access Systems" submitted by Laura Choy Lai Fun in partial fulfillment of the requirements for the degree of Master of Science

ABSTRACT

The focus of this research project is to consider the design and performance of turbo codes in the context of the W-CDMA system requirements. The design parameters of interest include the number of decoding iterations, the size of the interleaver, the selection of the type of interleaver, the appropriate selection of constituent codes with respect to the constraint length and code polynomials, and the choice between dual termination and single termination. The research shows that:

a) Larger interleavers provide greater randomness, which translates into better performance.

b) The interleaver has to provide the required randomness such that the encoded sequences are not correlated and it has to be capable of breaking up critical sequences.

c) Larger constraint length constituent codes do not necessarily result in better turbo codes.

d) There exists a maximum decoding iteration beyond which the performance does not improve and can deteriorate.

e) Joint termination method did not prove to be a better option compared to single termination method for small interleavers.

In conclusion, it is shown that the choice of parameters cannot be carried out in isolation, and they have to be chosen in conjunction with each other.

Acknowledgements

I would like to thank Dr. Ivan J. Fair and Dr. Witold A. Krzymien for introducing me to this project and the supervision throughout the course of this work. I would also like to acknowledge Zhai Feng Qin for helping me at TR*Labs* while I was doing this work.

The financial support offered by TR*Labs* and the University of Alberta is much appreciated.

Most of all, I would like to thank my parents and my grandmother for the support they have given me throughout my life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

P_e	first event error probability
P_b	Bit error rate
a_d	Number of pats with distance d from all zero path
E_b/N_o	Signal-to-noise ratio
R_c	Code rate
d_{free}	Free distance
d_{eff}	Effective distance
K	Constraint Length
$f_{D,n}$	Doppler shift
f_m	Maximum Doppler shift
f_c	Carrier frequency
λ_c	Wavelength of the arriving plane wave
τ	Path Delay
ψ	Phase Shift
θ	Angle of Incidence
φ_{xx}	Autocorrelation of x
φ_{xy}	Cross correlation of x and y
S_{xx}	Power spectral density
$p_z(x)$	Probability distribution of z
μ_{xx}	Autocovariance function

LIST OF ABBREVIATIONS

AMPS	Advanced Mobile Phone System
APP	A-Posteriori Probability
ARIB	Association of Radio Industries and Business
AWGN	Additive White Gaussian Noise
ASIC	Application Specific Integrated Circuits
BER	Bit Error Rate
CDMA	Code Division Multiple Access
CRC	Cyclic Redundancy Check
ETSI	European Telecommunications Standards Institute
FAX	Facsimile
FDD	Frequency Division Duplex
FER	Frame Error Rate
FLMPTS	Future Land Mobile Public Telecommunication System
GSM	Global System for Mobile Communications
IFFT	Inverse Fast Fourier Transform
IMT-2000	International Mobile Telecommunications-2000
IMT-MC	International Mobile Telecommunications-multi carrier
IMT-DS	International Mobile Telecommunications-direct spread
IS-95	International Standards 1995
ITU	International Telecommunications Union
JTACS	Japanese Total Access Communication System
LLR	Log Likelihood Ratio
LOS	Line Of Sight
MAP	Maximum A-Posteriori
NA-TDMA	North America- Time Division Multiple Access
NMT	Nordic Mobile Telephone
NTT-800	Nippon Telephone and Telegraph 800
PCCC	Parallel Concatenated Convolutional Codes
pdf	Probability Density Function
PIL	Prime Interleaver

RSC	Recursive Systematic Convolutional
S-random	Spread Random
SOVA	Soft Output Viterbi Algorithm
TACS	Total Access Communication System
TDMA	Time Division Multiple Access
TPC	Transmit Power Control
W-CDMA	Wideband Code Division Multiple Access

I. INTRODUCTION

The first generation of public cellular wireless networks was deployed in the late 1970s and early 1980s to provide voice telephony services to mobile subscribers over a wide area [1]. The Advanced Mobile Phone System (AMPS), the Nordic Mobile Telephone (NMT), the Total Access Communication System (TACS) and the Nippon Telephone and Telegraph-800/Japanese Total Access Communication System (NTT-800/JTACS) were the few first generation analog systems that were developed in different regions of the world. Mobile phones, however, were initially mainly the privilege of a small group of users because of limitations of the systems and their service area.

When digital transmission techniques were integrated into wireless systems, the second generation of wireless networks was born. These wireless networks were not only more robust than the previous generation systems, but they were also able to support both voice and low-speed data/facsimile (FAX) services with their increased capacity. Time division multiple access (TDMA) and code division multiple access (CDMA) are two of the multiple access schemes deployed in this generation. The second generation digital systems, such as Global System for Mobile Communications (GSM), North American TDMA (NA-TDMA), and CDMA-based IS-95, received wide acceptance and experienced unexpected growth rates.

Parallel to the growth of wireless telephony, the strong emergence of Internet services had intensified the demand for a third generation wireless system. In the late

1980s, the International Telecommunications Union (ITU) began a study of third generation communications systems known initially as FLMPTS (Future Land Mobile Public Telecommunication System), which later evolved into International Mobile Telecommunications-2000 (IMT-2000).

The ITU has agreed on five wideband radio interface standards under the IMT-2000 initiative. The third generation wireless system in North America will be an enhancement of IS-95, which is based on code division multiple access. This radio interface standard is known as IMT-MC (multi-carrier) due to its use of three carriers on the downlink. However, only the single carrier (1.25 MHz) version is in product development. The European Telecommunications Standards Institute (ETSI) and the Association of Radio Industries and Businesses (ARIB) in Japan have decided to adopt wideband direct sequence code division multiple access (W-CDMA) into their frequency division duplex (FDD) bands in their third generation wireless system. As a result, this radio interface is expected to become a global standard in the near future. For that reason, the focus of this thesis will be based on the IMT-DS (direct spread) system.

The requirements of the IMT-2000 system include support of wideband services at data rates as high as 2 MB/s, wide area coverage, and the ability to use multiple services simultaneously. To deploy these new features over various radio environments, third generation systems must offer a very flexible air interface as

well as improved spectrum efficiency compared to the second generation wireless systems.

A major additional feature of the third generation wireless systems is the ability to accommodate high speed data transmission. For speech transmission, a typical bit error rate (BER) of $P_b = 10^{-3}$ is acceptable, but for data transmission, error rates are required to be at least three orders of magnitude lower. To meet this performance requirement, powerful error control coding must be used.

There are two basic classes of error control coding, block and convolutional codes. Both codes map k information bits to n encoded bits, resulting in code rate of k/n. Unfortunately, the added bits result in increased bandwidth and lower energy per transmitted symbol.

This chapter will consider each of these codes briefly. Following that, concatenated codes and turbo codes are discussed briefly. The chapter then concludes with an overview of the main topic of the thesis.

1.1 Block Codes

Block codes are always transmitted in blocks of data that are n bits long. For a single n-bit long code vector, there exist 2^n possible combinations. Only 2^k possible code vectors, however, are used for transmission. If the decoder receives a corrupted vector, it will decode the correct word by evaluating the nearest valid code vector.

On the other hand, if the corrupted vector is a valid code vector for these codes, it will be undetected. For the corrupted code vector to become another valid code, the word must contain at least d_{min} errors. d_{min} is the minimum number of bits that must be changed to end up with another code vector. When code vectors differ by at least d_{min} positions, the block code decoder is guaranteed to correct up to t random symbol errors per word where [2]:

$$t \leq \frac{d_{\min}-1}{2} \tag{1}$$

As mentioned earlier, block codes map a k-bit information vector into an n-bit long coded vector. As n and k increase, coding by look-up tables becomes impractical. Therefore, linear block codes are usually implemented. A code is said to be linear if the addition of any two or more code vectors forms another code vector. This simplifies both coding and decoding of the codes. Cyclic codes such as Reed Solomon codes are examples of block codes that have a convenient structure and provide an easy way to encode and decode.

1.2 Convolutional Codes

Convolutional codes are usually described by the constraint length of the codes, K, and the code rate of the encoder, k/n. The code rate is the ratio of the number of information bits fed into the convolutional encoder to the number of output symbols produced by the convolutional encoder in a given encoder cycle. The convolutional encoder can be thought of as a linear finite state shift register. The shift register contains K-1 stages and each stage is k-bits long. The encoded sequence is produced

by n output linear algebraic function generators. Figure 1-1 shows an example of a convolutional encoder where k=1, n=2 and K=3.

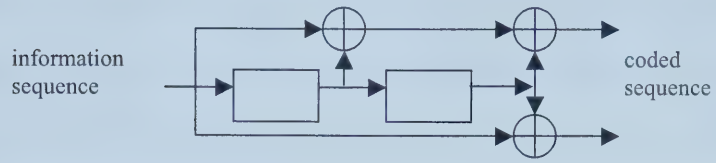


Figure 1-1: Convolutional encoder of rate ½ and contraint length K=3.

Unlike block codes, which have fixed length n, a convolutional encoder is a finite state machine. Therefore, decoding these codes involves a search for the most probable sequence. The correcting capability of a convolutional code depends on the differences between valid paths. This being said, it is more difficult to quantify the probability of error for the code.

Catastrophic convolutional codes have the potential to generate unlimited numbers of decoded bit errors in response to finite number of errors in the demodulated bit sequence. This is avoided with non-catastrophic convolutional codes which do not contain closed loop paths in their trellis diagrams with all-zero output other than the one taken with all-zero input.

1.3 Concatenated Code

A concatenated code consists of two codes that are combined to form a larger code. There are two basic classes of code concatenation: serial and parallel. In serial concatenation, the information stream is encoded by an outer code that is often a high rate systematic code such as a Reed-Solomon code. The coded data stream is

then fed into a second inner coder. The inner code is usually a low rate convolutional code. Convolutional codes are used because soft input decoders, such as Viterbi decoders, can decode them efficiently. However, these codes, alone, are usually unable to achieve the target bit error performance at sufficiently low signal-to-noise ratios. A second stage of decoding by the outer high rate block code, therefore, is required [3]. Reed-Solomon codes are used because of their ability to correct the bursts of bit errors that can result from typical trellis-based decoders.

1.4 Turbo Codes

A breakthrough in error control coding occurred in late 1993 when a paper by Berrou, Glavieux, and Thitimajshima [4] was published. The new codes, called turbo codes, achieve performance that approaches the theoretical limit derived by Shannon [5]. It has been shown that an the additive white Gaussian noise (AWGN) channel, rate ½ turbo codes reach a bit error rate of $P_b=10^{-5}$ at a signal-to-noise ratio of only 0.7 dB [4]. To achieve the same performance with conventional convolutional codes of the same rate and constraint length would require a signal to noise ratio of about 4.3 dB [4]. The unprecedented performance of turbo codes has resulted in their adoption in the IMT-2000 standards [6].

Although the structure of the parallel concatenated convolutional code (PCCC) based turbo codes is well established, the selection of their parameters is still not fully understood. The focus of the research in this thesis is to consider a number of parameters of interest for turbo codes in context of the IMT-2000 system

requirements. The parameters of interest include the number of decoding iterations, the size of the interleaver, the selection of the type of interleaver (the prime or the spread interleaver), the appropriate selection of constituent codes with respect to the constraint length and code polynomials, and the choice between dual termination and single termination. This research shows that the parameters of turbo codes cannot be selected in isolation, but need to be considered jointly.

1.5 Thesis Overview

The structure of turbo codes is discussed further in Chapter II. The chapter begins with a brief introduction to the building blocks of turbo codes such as its constituent codes, interleaver, decoding algorithm and termination methods. This chapter gives an overview of the parameters of concern for turbo codes such that a brief understanding for the codes is established.

Chapter III presents some basic information about mobile radio channel. The chapter starts with a summary of the characteristics of a mobile channel as taken from [7]. This is followed by a discussion in channel modeling and power control. The chapter concludes with a brief description of the simulation structure used for this thesis.

Chapter IV shows the simulation results for turbo codes with respect to their parameters and their discussion. Chapter V concludes the thesis with a summary of the findings of the research.

II. TURBO CODES

In his pioneering work, Shannon proved that the minimum required E_b/N_o to achieve an arbitrarily small probability of error over an infinite bandwidth additive white Gaussian noise channel is -1.6 dB [5]. This limit is called the Shannon limit. It has long been the goal of coding theorists to construct powerful error control coding that would be able to provide performance near this limit.

It is well known that increasing the constraint length of convolutional codes leads to better performance, however, the complexity of the decoding algorithms increases with length to the point where decoding becomes unrealizable. Similarly, the structures developed for block codes to date prevent the error correcting capability of these codes from achieving the limit [2].

Many proposals have been directed towards constructing powerful but practical codes with large equivalent block lengths by breaking the decoding algorithms into a series of simpler decoding steps. A prime example of this approach is concatenated coding. Forney was the first to show that it is possible to achieve large coding gains by combining two or more relatively simple component codes [3]. The serially concatenated codes he proposed have the error correction capability of longer codes, but have a structure that is relatively easy to decode.

In 1993, turbo codes were introduced by Berrou, Glavieux, and Thitimajshima [4]. These authors reported a required E_b/N_o of 0.7 dB for a bit error probability of

10^{-5} with turbo codes of code rate ½. This chapter gives a brief description of turbo code structure, followed by some basic background information on its basic component codes, convolutional codes and recursive convolutional codes. This chapter also gives an overview of the parameters of interest in turbo codes, such as its constituent code polynomials, common decoding algorithms, interleavers, number of decoding iterations, and trellis termination.

2.1 Turbo code structure

The first turbo codes proposed were parallel concatenated convolutional codes (PCCC) whose encoder is formed by two (or more) constituent systematic encoders joined through an interleaver with the structure shown in Figure 2-1. The input information sequence, X, is fed into the first encoder, and, after being interleaved, the interleaved information enters the second encoder. Finally, the resulting code word of the parallel concatenated code consists of the input bit to the first encoder, x_1, followed by the parity check bits of both encoders, x_2 and x_3, respectively. Various code rates may be achieved by means of puncturing or periodically deleting some

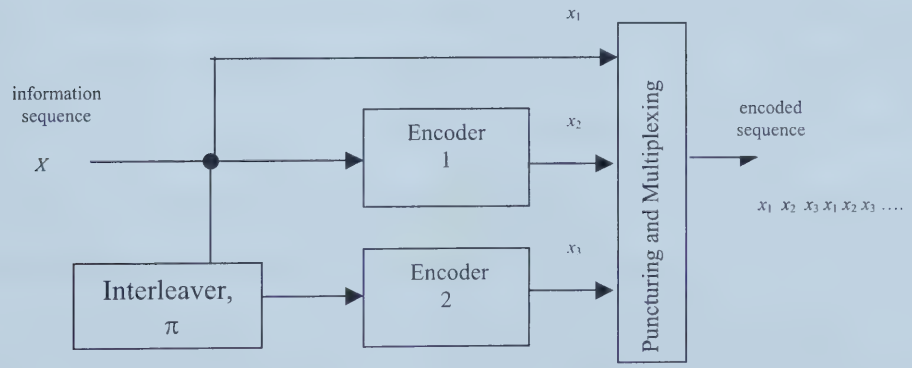


Figure 2-1: Turbo code encoder

coded symbols.

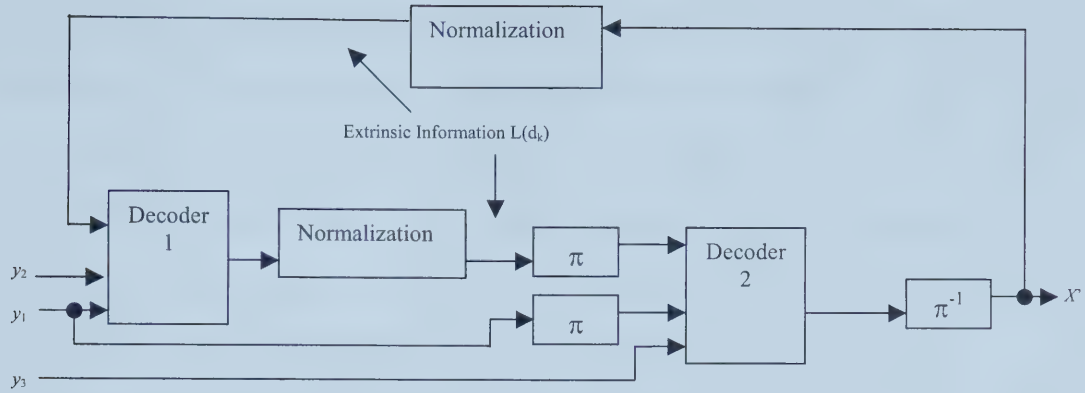


Figure 2-2: Turbo code decoder

The basic idea of a turbo code can be seen as breaking up the decoding of a fairly complex long code into a number of successive decoding steps while transferring soft information between the decoding steps with little loss of performance. The iterative decoder shown in Figure 2-2 involves the serial concatenation of decoders for the constituent codes, separated by the same interleaver, π, used in the encoder. As shown in Figure 2-2, the received sequence can be separated into three components, the received version of the information bit, y_1, the received version of the encoded bit of encoder 1, y_2 and the encoded bit of encoder 2, y_3. The constituent decoders in a turbo decoder learn from each other by exchanging soft decisions that are indicative of the reliability of the symbols. These soft decisions are the log-likelihood ratios (LLR) of the symbols.

At the first iteration, the decoding algorithm is applied to the first decoder with its received data, y_1 and y_2. Output soft decision information related to the symbol

includes the LLR of both the parity bits and the information bit of the symbol. Since the decoders have different parity bits, the only information that is useful to the second decoder is the LLR of the information bit, better known as the extrinsic information. The normalization structure extracts this extrinsic information, and the LLR values are interleaved before entering the second decoder. This extrinsic information is used as an a-priori estimate of the information symbols at the second decoder. Again, the soft decision output of this second decoder includes the LLR of the encoded symbol. However, this LLR not only includes the extrinsic information of itself and the LLR of the information bit but the extrinsic information of the previous decoder as well. This soft decision is deinterleaved and the decoding process is repeated.

The extrinsic information from the previous decoder and the demodulated data appearing as an input to the current decoder are correlated [8]. To decrease the correlation, it is important not to feed a decoder with information that stems from itself. The normalization block extracts the extrinsic information from the soft decision and uses it as an a-priori estimate of the information symbol for the first decoder. This iterative decoding process continues until the predefined stopping criterion is reached. At this point, the deinterleaved soft decision sequence from the second decoder. X' will be the decoded version of information sequence, X.

Statistical dependencies between the soft decisions of the second decoder and the first decoder may be encountered. When the reliabilities of these soft decisions

are high, statistical dependency may be an issue. As the number of decoding iterations increases, the output of the constituent decoders generally increases in accuracy, however, the constituent decoders will become more dependent on the extrinsic values. Since the extrinsic information between the decoders becomes correlated as the number of decoding iterations increases, the improvement in performance from the current iteration to the next becomes marginal.

2.2 Component Codes – Convolutional codes

Convolutional codes are generated by passing information through a shift register and performing an algebraic function on the contents of the shift register to produce a coded sequence. The algebraic function and the length of the shift register are best described by a generator polynomial, which is often given in octal form. For example, a generator polynomial of 7 ≡111 indicates that the first three taps are added using modulo-2 arithmetic. The structure below shows a convolutional code of code rate ½, K=3 with generator polynomial of (7,5). In this figure, the block defines a unit delay element:

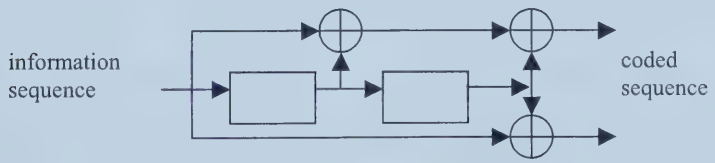


Figure 2-3: Convolutional code structure for (7, 5)

The length of the shift register, $(K-1)$, is determined by the constraint length of the code K. For example, the constraint length for 7 is 3. The convolutional encoder can be thought of as a finite state machine where the bits of the shift register will define the state of the encoder. The number of states of the convolutional encoder will

depend on the memory length of the shift register. For example, in Figure 2-3, the encoder has a memory length of 2 and the encoder can be in any of the 4 states {***00***,***01***,***10***,***11***}. Depending on the current state of the encoder, the encoder will move to a predefined next state depending on the input bit.

Table 2-1 and Figure 2-4 show the next state table and the trellis diagram of the convolutional code (7,5). These are two alternative ways of describing convolutional codes. Assuming the encoder is initially in the all-zero state, if the first input bit is a 0, the encoder will be in the ***00*** state with an output sequence of ***00***. Conversely, if the first input bit is a 1, the encoder will be in ***10*** state with an output sequence of ***11***. The dashed and the solid lines in the trellis diagram represent the input bit into the encoder being 1 and 0, respectively. Any coded sequence that would return the encoder to an all-zero state is considered a valid sequence. Note that an all-zero input sequence will always output an all-zero sequence, therefore, the all-zero output sequence is always a valid sequence.

Assuming an all-zero sequence as the information sequence, any other valid sequence beside the all-zero sequence will be a corrupted sequence. The 1's in these corrupted sequences are denoted as errors, and the total number of 1's determines the Hamming distance of the code. The free distance, d_{free}, is the minimum number of errors in a decoded sequence.

For the (7,5) code shown in Figure 2-4, it can be found that the free distance, d_{free}, for this code is 5 with an input weight of 1. This also means that in order to have a single decoding error, there need to be at least 5 errors in the decoding process.

Current State, S_o		M, information bit	Next State, S_f		Coded bits
00	X_a	0	00	X_a	00
		1	10	X_c	11
01	X_b	0	00	X_e	11
		1	10	X_c	00
10	X_c	0	01	X_b	10
		1	11	X_d	01
11	X_d	0	01	X_b	01
		1	11	X_d	10

Table 2-1: Next State Table for a (7,5) Convolutional Code

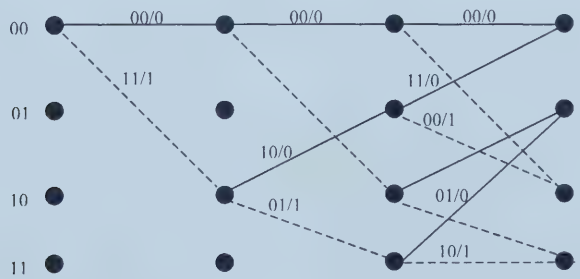


Figure 2-4: Trellis diagram for convolutional code (7,5)

The Hamming distances of all valid sequences can also be found from their transfer function. The current states, next states, information bits, and encoded bits can be described in an algebraic form. The state diagram shown in Figure 2-5 can be used to derive their transfer function. The all-zero state is split into two states, one of which represents the input and the other the output of the state diagram. The self-loop at state X_a is eliminated since it contributes nothing to the distance properties of a code sequence. The ratio of the final state over the start state, X_e / X_a will uniquely describe all possible valid sequences.

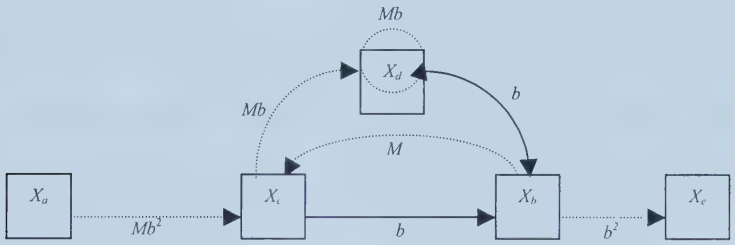


Figure 2-5: State diagram for code rate ½, *K*=3, convolutional code (7, 5).

Taking the above example:

$$\begin{aligned} X_b &= bX_c + bX_d \\ X_c &= MX_b + Mb^2X_a \\ X_d &= MbX_c + MbX_d \\ X_e &= b^2X_b \end{aligned} \tag{2}$$

Describe each state in terms of X_d :

$$\begin{aligned} X_b &= \frac{X_d}{M} \\ X_c &= \frac{(1-Mb)}{Mb}X_d \\ X_a &= \frac{(1-2Mb)}{M^2b^3}X_d \\ X_e &= \frac{b^2}{M}X_b \end{aligned} \tag{3}$$

The transfer function will be :

$$\frac{X_e}{X_a} = \frac{b^5 M}{1 - 2bM} = b^5 M + 2b^6 M^2 + 4b^7 M^3 + \cdots \tag{4}$$

The exponent of b indicates the number of 1's in the encoded sequence for that path from the all-zero sequence, the exponent of M denotes the number of 1's in the information sequence from that path, and the coefficient of each term represents the number of paths that would be decoded in this manner. These coefficients are better known as the multiplicities of the error sequences [9]. In (4) the first term states that to make a single decoding error, one needs at least 5 errors in that particular decoded sequence; similarly, the second term notes that to make two errors, one needs to decode a sequence with at least 6 errors. Although the transfer function method seems more efficient, the complexity of the algebra grows exponentially with the constraint length of the code.

The distance can be related to the upper bound on the first event error probability of the code in the form [2]:

$$P_e \leq \sum_{d=d_{free}}^{\infty} a_d Q\left(\sqrt{\frac{2\varepsilon}{N_o} R_c d}\right). \tag{5}$$

P_e is the first event error probability of the code, a_d is the number of paths with distance d from the all-zero path and $Q(.)$ is the Q function, where:

$$Q\left(\sqrt{\frac{2\varepsilon}{N_o} R_c d}\right) \leq e^{-\frac{\varepsilon}{N_o} R_c d}, \tag{6}$$

ε / N_o denotes the signal-to-noise ratio of the bit and R_c is the code rate. Unlike block codes, the error performance of convolutional codes depends on the code sequences and trellis structure. This bound can therefore be used as an approximation of the error performance of convolutional codes. In general, the constraint length is proportional to minimum distance [2]. For that reason, it is best to use codes with large constraint lengths in order to ensure large free distance. Now, two specific types of convolutional codes are reviewed: systematic convolutional codes and recursive convolutional codes.

2.3 Systematic convolutional codes

Systematic convolutional codes are convolutional codes where the information sequence is included as part of the encoded sequence. Usually, the transmission of the information sequence does not help to increase the minimum distance. In the example depicted in Figures 2-6 and 2-7, to make a single error with the systematic code, the path needs to contain four channel errors. However, the non-systematic code shown in Figure 2-4 requires five channel errors. In this case, the systematic codes performs worse than the non-systematic code.

It has been established that the error correcting capability of convolutional codes is closely related to their d_{free}. Optimal generators for binary short constraint length codes for several code rates had been tabulated in [2]. They are optimal in the sense that for a given constraint length, these codes provide the largest possible d_{free}.

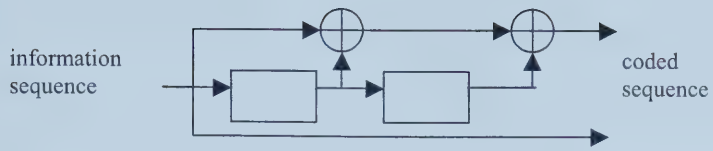


Figure 2-6: Systematic convolutional code structure for 7

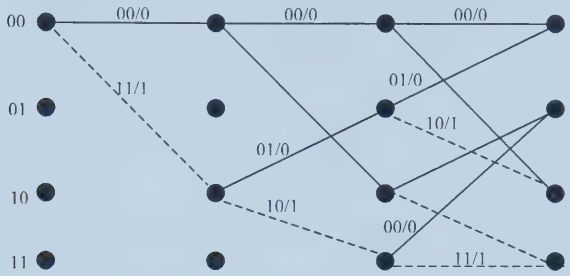


Figure 2-7: Trellis diagram for systematic convolutional code 7

2.4 Recursive convolutional codes

With the convolutional codes discussed so far in this thesis, it is possible to force the decoder to return to an all-zero state, regardless of what state it is in, by flushing in zeros into the input. This is called "terminating" the sequence. One consequence of a terminated sequence in convolutional codes is that an information sequence containing a single error will always correspond to a valid sequence through the trellis. This motivates the use of recursive convolutional codes because there is no single error sequence for these codes.

Recursive convolutional codes are described by two vectors, one describing the feedback connections which affect the state of the encoder, and the other describing the feedforward connections that generates the coded sequence. Figure 2-8 depicts a recursive convolutional code encoder. Unlike non-recursive convolutional codes, the state of a recursive convolutional encoder depends on both the information sequence

and the feedback connections. Therefore, with an input sequence of weight one, the trellis for these codes is unable to return to the all-zero state. In other words, a single error sequence does not exist for these codes.

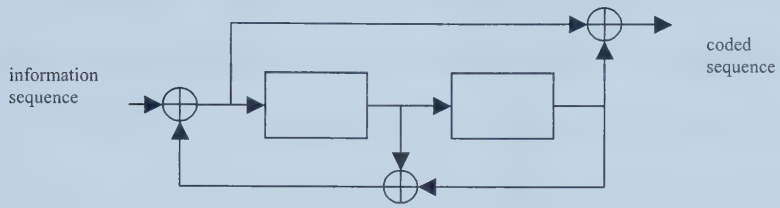


Figure 2-8: Recursive convolutional encoder of (7, 5)

Since there are no single error sequences, weight-2 error sequences are the most likely error sequences to be decoded in situations where channel errors are independent and the signal-to-noise ratio is relatively high. In these situations, weight-2 error sequences dominate the error performance bound. The likelihood of decoding a weight-2 sequence decreases with its length. Therefore, to maximize the length of 2-error sequences, it has been suggested that the feedback polynomial be a primitive polynomial [10].

Recursive systematic convolutional (RSC) codes are a subclass of recursive convolutional codes. The encoded bits consist of the input bit and one or more bits encoded through a recursive convolutional code structure. These codes are used as the component codes of turbo codes and will be discussed in the following section.

2.5 Constituent code polynomials- Recursive Systematic Convolutional (RSC) codes

The use of recursive systematic convolutional codes as constituent codes for turbo codes is considered here. Since recursive convolutional codes are linear codes, the

conventional measures of performance for convolutional codes apply to the performance of each constituent code.

In general, the goal when designing the best component codes for turbo codes is to maximize the minimum distance corresponding to 2-error sequences, better known as the effective free distance, d_{eff}. This is because, it is argued, that the code performance of turbo codes at a region where it is most effective is determined largely by the weight among codewords corresponding to input words of weight-2 [11]. Also, it is argued that the distance is very large for input words of weight-3 and above. However, at low E_b/N_o, optimizing the weight distribution of the codeword is more important [12]. This can be seen in equation (4). Given a fixed turbo code size, the multiplicity of the sequences corresponding to weight 2 and above is fixed. At low E_b/N_o, the multiplicity of codewords corresponding to weight-3 and above is not large enough to be ignored. Therefore, the weight distribution of the codeword is important.

An advantage that turbo codes have over conventional convolutional codes is that both constituent decoders must jointly decode a sequence in error before the decoder outputs an error. Since only information about data bits is exchanged between the constituent decoders, the most likely sequences to be decoded in error are low weight data sequences that are mapped to low weight encoded sequences by both constituent encoders [13].

It is straightforward to verify that in RSC codes, weight-2 data sequences that generate low weight encoded sequences are sequences in which the two logic ones are separated by P-1 zeros, where P is the period of repetition of the feedback polynomial. With primitive feedback polynomials, the shortest length of the weight two sequence of concern is P+1 bits, where $P=2^{K-1}-1$ and K is the constraint length of the code. Also, weight–two sequences of length ($nP+1$), when n is an integer, also generate low weight encoded sequences. However, the weight of the output sequence increases with n. For example, for a constraint length $K=4$ code with a primitive polynomial, $P=2^{4-1}-1=7$, the shortest weight-two sequence is 1 000 000 1, the next shortest is 1 000 000 0 000 000 1, etc. In this thesis, these weight two sequences will be referred to as critical sequences.

The importance of weight-2 data sequences in turbo decoding is clear, as mentioned earlier. The distance that contributes to weight-2 data sequences is called the effective free distance. Thus, when selecting constituent codes for turbo codes, effort is usually made to maximize the effective free distance of the constituent RSC codes rather than maximizing the free distance, as in conventional convolutional codes.

Benedetto et al. have tabulated codes with large distance properties in [9] and they are used as the source of comparison in this simulation. The search method used in [9] first finds codes that maximize the minimum distance d_i that is generated by low weight input sequences and, as a secondary criterion, minimizes the number of

code sequences with that minimum distance. As a result, the search emphasized maximizing the minimum weight of the code sequences generated by low weight input sequences.

It is well known that the performance of convolutional error control codes increases with constraint length [2]. This result is based on the observation that in well-designed codes, the free distance increases with constraint length, and in stand-alone decoding a larger free distance results in better decoding performance. It has not been proven if a similar relationship holds concerning the performance of iterative decoding. It could be expected that larger constraint length codes, however, will result in superior performance when embedded as constituent codes in a turbo code.

Then again, the performance of a turbo code is also closely linked with the ability of the interleaver to break up critical sequences. Larger constraint length codes tend to have longer critical sequences, therefore the interleaver may not be able to adequately break up such critical sequences, leading to a reduction in decoding performance.

2.6 Constituent Decoding Algorithm

In order to facilitate iterative turbo decoding, the decoding algorithms for the constituent codes must produce soft decisions regarding the likelihood of information symbol values. The Maximum A- Posteriori (MAP) algorithm [14]

provides the logarithm of the ratio of the a posteriori probability (APP) of each information bit being 1 to the APP of it being 0 and is the usual choice for decoding turbo codes. However, this algorithm is somewhat complex to implement because of the numerical representation of probabilities and the number of mixed multiplications and additions [15].

Some approximations of the MAP algorithm have been proposed to simplify implementation. These include the soft-output Viterbi Algorithm (SOVA) [16], Log-MAP Algorithm [17] and the Max-Log-MAP algorithm [18]. Since the above mentioned algorithms are processed in the logarithmic domain, multiplications and divisions will be reduced to easier to handle additions and subtractions. Max–Log-MAP and SOVA offer identical performance if the algorithm considers only hard decisions, but the performance is inferior to decoding by soft information. Also, when using soft information, the SOVA and Max-Log-MAP algorithms differ.

Assuming binary decoding, Max-Log-Map looks at two paths on the trellis per step, the best path with bit zero, and the best path with bit one at each transition. One of these paths will always be the maximum likelihood path. On the other hand, SOVA will only find one of these two paths and not necessarily the other. Thus, the other path may have been eliminated before merging.

The difference between the Max-Log-Map algorithm and Log-Map algorithm is the approximation used in calculating the logarithm. For an exact calculation of

$\ln\{\exp(\delta_1) + ... + \exp(\delta_n)\}$ used in the Log-Map Algorithm, the Jacobian logarithm is used,

$$\begin{aligned} \ln(e^{\delta_1} + e^{\delta_2}) &= \max(\delta_1, \delta_2) + \ln(1 + e^{-|\delta_1 - \delta_2|}) \\ &= \max(\delta_1, \delta_2) + f_c(|\delta_1 - \delta_2|) \end{aligned} \tag{7}$$

where $f_c(.)$ is the correction function. In the Max-Log-Map algorithm, only the first term of equation (7) is used. As one can see, the difference between the two algorithms is the correction function. The expression $\ln\{\exp(\delta_1) + ... + \exp(\delta_n)\}$ is computed recursively and can be computed exactly. Supposed $\delta = \ln[\exp(\delta_1) + ... + \exp(\delta_{n-1})]$ is known and $\Delta = \exp(\delta_1) + ... + \exp(\delta_{n-1})$.. Hence,

$$\begin{aligned} \ln(e^{\delta_1} + \cdots + e^{\delta_n}) &= \ln(\Delta + e^{\delta_n}) \\ &= \max(\ln \Delta, \delta_n) + f_c(|\ln \Delta - \delta_n|) \\ &= \max(\delta, \delta_n) + f_c(|\delta - \delta_n|) \end{aligned} \tag{8}$$

By introducing the correction factor at each step, the results will be identical to the original MAP algorithm. However, by calculating $f_c(.)$, the simplification of the Max-Log-Map is lost. Since correction only depends on $|\delta_2 - \delta_1|$, a one-dimensional pre-computed table is used, and the performance is similar to Log-Map. It is shown in [17] that SOVA requires 0.7 dB higher in signal-to-noise ratio than Log-Map in order to achieve a bit error rate of $10^{-4}$and Max-Log-Map lies roughly in between.

The following table is the correction table used in the simulation.

$X=\|\delta_2-\delta_1\|$	$f_c(\|\delta_2-\delta_1\|)$
$0<x\leq 0.5$	0.5
$0.5<x\leq 1.0$	0.387
$1.0<x\leq 2.0$	0.210
$2.0<x\leq 3.0$	0.0845
$3.0<x\leq 4.0$	0.00335
$x>4.0$	0.0

Table 2-2: Correction Table.

2.7 Interleavers

The performance of an iterative decoder depends in part on the correlation of the sequences at the input to the constituent decoders [8]. Since correlation of these input sequences is a function of the interleaver, the design of the interleaver has a direct impact on the performance of turbo codes.

In general, a good interleaver should map received sequences that cannot be decoded correctly by one constituent decoder to sequences that can be decoded correctly by the other constituent decoder. This is achieved if the interleaver randomizes the input sequence such that the parity sequences generated by the two decoders are as diverse as possible.

Constituent codes of turbo codes that have low minimum distances (if unpermuted) due to high weight input sequences are often superior to other codes

with higher unpermuted minimum distances that are caused by low-weight input sequences [19]. However, low weight input sequences often generate low weight output sequences. As a consequence, if a low weight input sequence generates a low weight output sequence for the first encoder, the input sequence should be interleaved such that the other constituent encoder encodes a high weight output sequence.

A number of interleavers have been proposed to be embedded in turbo code structures. The simplest interleaver is the uniform block interleaver [2]. Also, several non-uniform interleaver structures have been proposed and analyzed in the literature [13,20]. The prime interleaver is described in [20] and is currently proposed for IMT-2000 (IMT-DS) systems [6]. Also of interest is the spread random interleaver [13]. These interleavers will be discussed briefly below.

Uniform block interleaving is the simplest interleaving algorithm. Information is written into the block interleaver row by row and read out column by column. Because of the recursive nature of RSC codes, an interleaved version of a sequence of input weight two is often not a valid sequence. The minimum distance of the code, however, usually corresponds to an input sequence of weight 3 and above. These low weight input sequences often generate low weight output sequences. With uniform block interleaving, these interleaved low weight input sequences are unlikely to generate a higher output code sequence. Non-uniform interleaving must therefore be employed.

The prime interleaver is an algorithmic interleaver based on prime numbers. Similar to uniform interleaving, the input sequence is written into a block of memory row by row and is read out column by column. However, permutations are performed on both rows and columns. Let an N by M memory space define the size of a mother interleaver that can be pruned to construct an interleaver of the desired size ranging from 257 to 8192 [20]. N can be 10 or 20, depending on the size of the desired interleaver, and M is chosen to be the prime number that results in a mother interleaver that is closest to, but not smaller than, the size of the desired interleaver.

Inter-row permutation is fixed depending on the interleaver size. Intra-row permutation uses an algorithmic computation depending on a primitive root. The specific primitive root that is used depends on the size of the mother interleaver. Inter-row permutation is performed to avoid the mapping of low input weight sequences to generate low output weight sequences by both constituent encoders while intra-row permutation satisfies the randomization of data sequences. Advantages of the prime interleaver include its flexibility to accommodate various block sizes and the ease with which the interleaving algorithm can be implemented.

The spread random (S-random) interleaver is an enhanced version of a purely randomly generated interleaver [13]. The S-random interleaver is based on the random generation of N integers from 1 to N with a constraint that ensures certain distance properties among the permuted bit positions. In contrast to the purely

random interleaver, each randomly generated permuted position for the S-random interleaver is compared with S_1 previously selected integers. If the current selection is within a distance S_2 from the S_1 previously selected integers, it is thrown back into the pool and another number is selected. Both S_1 and S_2 should be chosen to be larger than the memory of the two constituent codes. If both constituent codes are equal, it is appropriate to select $S_1=S_2$ [13]. It is not guaranteed that the selection process will converge and as the two parameters increase in size, the search time increases. It has been suggested that for an S-random interleaver of size N, selecting $S_1=S_2=\lfloor\sqrt{N/2}\rfloor$, offers a reasonable tradeoff between the randomness of the interleaver and the search time [13].

Although spread random interleavers exhibit good randomness properties, they must be constructed prior to run-time and must be stored in a look-up table. Different look-up tables must be constructed for each interleaver size of interest. As indicated above, there are many low-weight input sequences that produce low weight output sequences, and with a random interleaver, these sequences are much more likely to be broken up or generate a higher weight output sequence in the other encoder.

Regardless of the type of interleaver, it is expected that the performance of a turbo code will improve as the size of the interleaver increases since the effective randomness of the interleaver increases with size [13].

2.8 Number of decoding iterations

As discussed earlier, in iterative decoding, fewer errors typically remain after each iterative step [4]. It has also been shown that after several iterations, further improvement in performance become negligible and in some practical implementations, the performance can decrease due to the accumulation of numerical errors [21]. Since decoding delays increase with the number of iterations, determination of the number of allowed decoding iterations becomes an effort in determining the appropriate tradeoff between performance and decoding delay.

The simplest approach to stopping the iterative decoding process is to select the number of iterations for which every frame of data that will be processed. The drawbacks of this method are that it might not have reached an optimum result or the optimum result was reached before the fixed number of iterations. The latter results in a decoding delay that may not be desired. Alternatively, an upper bound to the number of iterations can be set, and techniques that stop the iterative process prior to this number of iterations, should the frame be determined to be without errors, can be deployed. These techniques are known as early stopping techniques. For example, in [20] it is proposed that CRC encoding be introduced before turbo encoding, and CRC decoding be performed following each constituent decoding stage. If the frame is determined to be without errors prior to reaching the final iteration stage, iteration is stopped, thus reducing the average number of decoding iterations. Other stopping criteria have been reported in the literature and they include those based on a variance estimate [22], monitoring the mean of the absolute values of the component

decoder output LLR's over a frame [21], etc. In simulations in this thesis, the decoder will stop iterating by checking the transmitted information sequence if an error-free frame is achieved, or it will continue for the fixed maximum number of decoding iterations.

2.9 Trellis termination

Trellis termination provides the decoder with additional information that results in superior decoding for packet-based systems. Trellis termination involves the insertion of redundant bits at the end of each frame of data to force the encoder back to the original all-zero state. In non-recursive codes it is sufficient to append (K-1) zeros to the end of each frame to drive the encoder to the all-zero state; in recursive codes this is not sufficient.

For a single recursive code, the termination sequence depends on the state of the encoder after N information bits. Figure 2-9 shows a method to terminate the trellis at the end of the block. Here N is the number of information bits. The switch is originally at position '*A*' for N cycles, and switches to position '*B*' for *(K*-1*)* cycles which will flush the encoder with zeros.

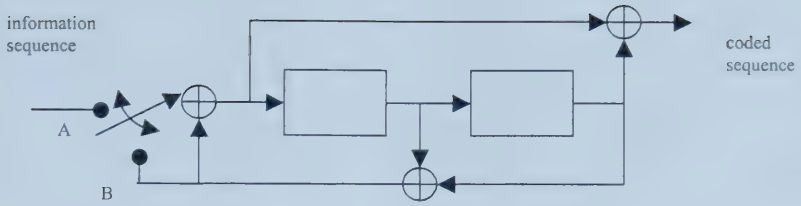


Figure 2-9: Trellis termination

For turbo codes, complete termination is more difficult. As turbo codes are embedded with an interleaver, it is highly unlikely that the interleaved data would terminate the second encoder. Therefore, the final state of the second encoder is undefined. Nonetheless, it is expected that termination of one or both of the encoders will help the decoding process.

There are two ways of terminating both encoders. One is to terminate both encoders with individual tail bits, as shown in Figure 2-10. With this method, the extrinsic information associated with these tail bits is not exchanged between constituent decoders during iterative decoding since the tail bits are different. This might affect the iterative decoding process since it depends on the soft decision of the other decoder. The other approach is to find a set of positions in the input data that can force the final state of both encoders to the zero state [23]. In this thesis, this second approach will be called joint termination.

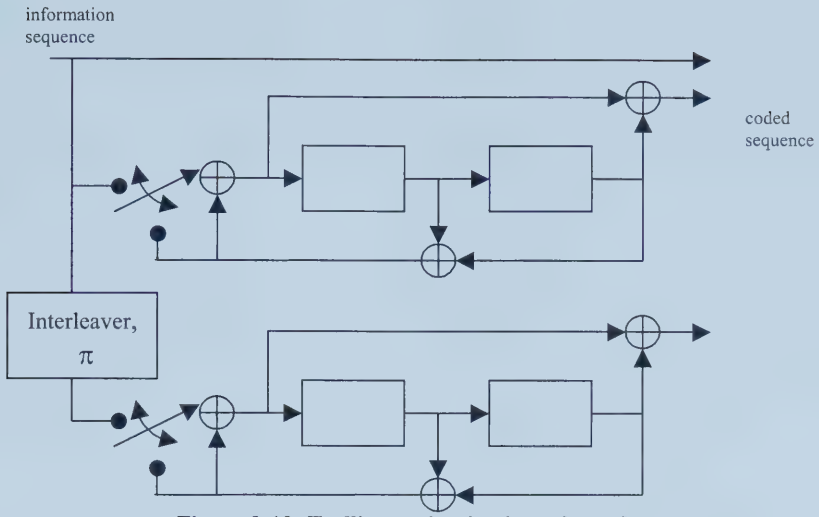


Figure 2-10: Trellis termination in turbo codes

Joint termination can be explained as follows [23]. Consider the linear map from the input data to the state of the first recursive convolutional code of memory m_1. The state vector is given by the m_1 vector representing the contents of the first encoder. Let N data bits be transmitted in every frame, and let Q be the number of data bits that are transmitted including $N + m$ tail bits. Then there exists an Q by m_1 matrix, $\boldsymbol{L}_1$, such that the state after encoding Q bits, $\boldsymbol{v}_1$,is given by

$$\boldsymbol{v}_1 = \boldsymbol{x}\,\boldsymbol{L}_1 \tag{9}$$

where $\boldsymbol{x}$ is a length-Q vector representing the data. Similarly, for the second recursive convolutional coder, the state after encoding the Q permuted data bits is given by

$$\boldsymbol{v}_2 = \boldsymbol{x}\boldsymbol{L}_2\boldsymbol{P} \tag{10}$$

where $\boldsymbol{P}$ is a Q by Q permutation matrix representing the interleaver. The overall total state of the turbo encoder is then given by

$$\boldsymbol{v}=[\boldsymbol{v}_1\ \boldsymbol{v}_2]^T = \boldsymbol{x}\boldsymbol{L} \tag{11}$$

where

$$\boldsymbol{L}=[\boldsymbol{L}_1\ \boldsymbol{L}_2\boldsymbol{P}] \tag{12}$$

is an Q by m_1+m_2 matrix representing the linear map from the input data to the final total state. Let n be equal to m_1+m_2; this will be the rank of the $\boldsymbol{L}$ matrix for most encoders of interest. Any n linearly independent columns of $\boldsymbol{L}$ span the range of the linear map[1]. Consequently, if the values of the n inputs are appropriately selected to these columns of $\boldsymbol{x}$, it is possible to force the final state to zero.

Now, the actual transmitted data per frame would be $N+n$. In general, the last n columns of the linear map $\boldsymbol{L}$ are not linearly independent, but the last m_1 columns of the linear map will always be linearly independent of each other. Therefore, the task will be to find the remaining $n-m_1$ columns from the linear map $\boldsymbol{L}$, which have to be linearly independent from the last m_1 columns. From the equation above, $\boldsymbol{L}$ is a linear mapping of impulse vectors at different positions across length-$\boldsymbol{Q}$, to the final states of the encoder after encoding these sequences, and is easily found by sending an impulse response and observing its final state.

As described above, n independent rows that span the linear map $\boldsymbol{L}$ must be found. Let these n rows form the matrix $\boldsymbol{S}$; the positions from which these n columns are taken will be the positions of the terminating bits. The overall final state can be expressed as:

$$\boldsymbol{v}=\boldsymbol{x}_a\boldsymbol{L}+\boldsymbol{x}_t\boldsymbol{S} \tag{13}$$

[1] Any column in the linear map can be expressed as linear combination of d, where d is the selected linearly independent n columns.

where x_a is the vector containing the input information bits with logic zeros in the positions of the terminating bits, and x_t is the vector containing the terminating bits. Requiring the overall from state to be zero dictates that the termination bits have value:

$$x_t = (x_a L)S^{-1} \tag{14}$$

An example will clarify the joint termination process. Consider the parallel concatenation of two identical constituent encoders with generator polynomials $(7,5)$[2], and given $Q = 10$, and an interleaver with the following permutation table:

$$\pi = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\ 3 & 2 & 1 & 5 & 10 & 4 & 6 & 8 & 7 & 9 \end{pmatrix}$$

$$L = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

Since the 4^{th}, 5^{th} , 9^{th} and 10^{th} rows are linearly independent, they can be selected to form the matrix S.

$$S = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix}, S^{-1} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \end{bmatrix}$$

[2] Note that the state of the encoder depends on the feedback vector 7 and the input information bits.

Terminating bits for any data vector can now be calculated. For example, consider the vector $\boldsymbol{x}_a$ = [1 0 0 0 0 1 0 0 0 0]. In this vector, the underlined bits are the logic zeros inserted into the positions of the termination bits. Then:

$$\boldsymbol{x}_t = [1\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0]\boldsymbol{L}\boldsymbol{M}^{-1}$$

$$= [1\ 0\ 1\ 0]$$

Therefore, the vector of data and terminating bits that would guarantee termination of both encoders is

$$\boldsymbol{x}_{new}=[1\ 0\ 0\ 1\ 0\ 1\ 0\ 0\ 1\ 0]$$

Multiplication of $\boldsymbol{x}_{new}$ by $\boldsymbol{L}$ verifies that the overall final state is zero.

III. STRUCTURE OF THE SIMULATED SYSTEM

The simulated model used in this thesis is based on the transceiver proposed for the IMT-DS system. Figure 3-1 shows a simplified block diagram of a mobile transmitter and a base station receiver.

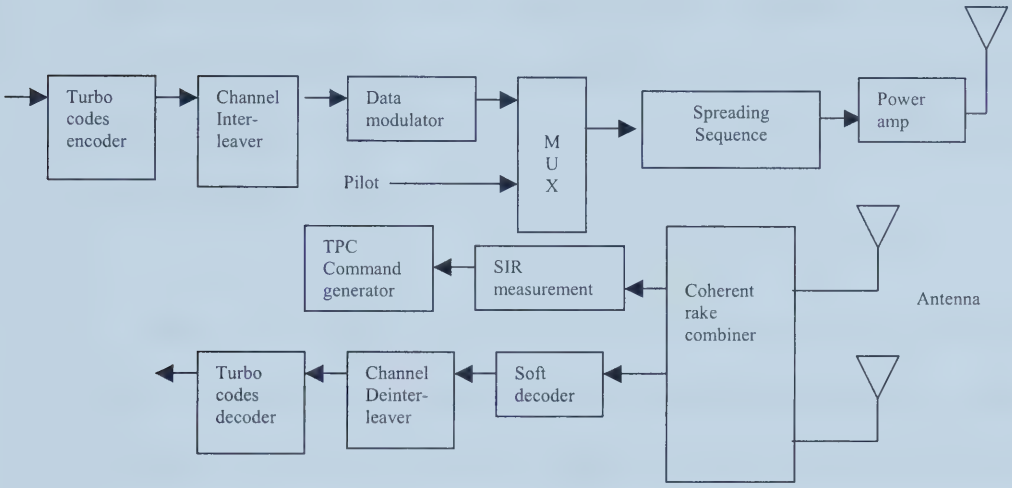


Figure 3-1: Simplified block diagram of a mobile transmitter and a base station receiver.

The proposed system uses turbo codes encoder to encode the user data, and the encoded sequence is channel interleaved and modulated before it is multiplexed with the pilot symbols and TPC command bits. Each frame consists of sixteen slots where each slot length is 1.25 ms [24]. Each slot will contain information symbols and pilot symbols.

These signals are spread by a Walsh/Gold sequence and multiplexed with codes from other users. This spreading procedure will ensure orthogonality with codes from other users. Following that, the combined signal is spread again by long random sequence and amplified before it is transmitted on the 2 GHz-frequency band.

At the base station receiver, two spatially separated antennas receive the spread signal transmitted by the mobile station. The multipath signal received by each antenna is matched filtered and then resolved into multiple versions of the transmitted signal by the rake receiver. These signals are demodulated, de-interleaved, and decoded. A 2-path equal power Rayleigh fading channel profile is assumed. Two spatially separated antennas with two finger Rake receivers at each antenna are used at the base station. Ideally, this is equivalent to four independent Rayleigh fading of equal gain.

Outputs of the rake receiver are then used for SIR measurements to drive fast TPC. If the measured SIR is lower or higher than the desired SIR, power control bits will be generated to inform the mobile station to increase or decrease the transmit power by a predefined step size. The use of rake combining and space diversity in the system minimizes the fluctuation of the channel attenuation and improves tracking capability for channel estimation.

To model the above system, basic characteristics of a wireless channel such as fading and multipath, must be well understood.

In urban areas, the heights of the mobile antennas are usually below the height of surrounding structures. With this setting, there is usually no single line of sight (LOS) path to the base station and multiple copies of the signal are received through reflections, scattering and diffraction. As a result, the signals arrive from different

directions with different delays. This phenomenon is called multipath propagation. A reasonable modeling assumption in urban areas is that the plane waves arrive from all directions with equal probability, which is known as isotropic scattering.

Small changes in the differential delays due to the mobility of the receiver cause large changes in the phases of the arriving plane waves. These phase differences manifest themselves as large variations in the amplitude and phase of the composite signal. The rapid fluctuations of the amplitude of the signal are known as fading.

This chapter begins with a summary of propagation models for radio channels taken from [7] and provides a discussion of channel simulation techniques used for radio link analysis. The next section gives a general discussion on power control, a technique used to minimize co-channel interference, the limiting factor of wireless systems' channel capacity. This is followed by a description of the channel model used for the research and concludes with an illustration of the resulting simulation test bench.

3.1 Flat fading

Consider a receiver moving along the x-axis with velocity v on a horizontal x-y plane. Transmitted signals are vertically polarized and, therefore, the electric field vector is aligned with the z-axis. The n^{th} plane waves arrives at the receiver with an angle of incidence $\theta_n(t)$. The receiver's movement introduces a Doppler shift into the incident plane wave given by

$$f_{D,n}(t) = f_m \cos\theta_n(t)$$

(15)

where $f_m = v/\lambda_c$ and λ_c is the wavelength of the arriving plane wave. Consider the transmission of a bandpass signal,

$$s(t) = \mathrm{Re}\{u(t)e^{i2\pi f_c t}\} \tag{16}$$

where $u(t)$ is the complex low-pass signal, f_c is the carrier frequency, and $\mathrm{Re}\{z\}$ denotes the real part of z. If the channel is comprised of N paths, the received bandpass waveform is

$$x(t) = \mathrm{Re}\{r(t)e^{i2\pi f_c t}\} \tag{17}$$

where the received complex low-pass signal $r(t)$ is given by

$$r(t) = \sum_{n=1}^{N} \alpha_n(t) e^{-j\psi_n(t)} u(t - \tau_n(t)) \tag{18}$$

where

$$\psi_n(t) = 2\pi\{(f_c + f_{D,n}(t))\tau_n(t) - f_{D,n}(t)t\} \tag{19}$$

is the phase associated with the n^{th} path and $\alpha_n(t)$ and $\tau_n(t)$ are the amplitude and time delay, respectively, associated with the n^{th} path, and $\psi_n(t)$ is the phase associated with the n^{th} path. From (18), the channel can be modeled by a time-variant linear filter having the complex low-pass impulse response

$$c(\tau, t) = \sum_{n=1}^{N} \alpha_n(t) e^{-j\psi_n(t)} \delta(\tau - \tau_n(t)) \tag{20}$$

where $c(\tau,t)$ represents the channel response at any time t to an impulse applied at time t-τ, and $\delta(.)$ is the dirac delta function.

Since $f_c + f_{D,n}(t)$ is very large, a small change in path delay $\tau_n(t)$ causes a large change in the phase $\psi_n(t)$. $\alpha_n(t)$ depends on the cross sectional area of the n^{th} scatterer but does not change significantly over small spatial distances. Therefore,

fading is primarily due to time variations of the random phases $\psi_n(t)$ that are caused by $f_{D,n}(t)$.

For narrow-band signals where the signal bandwidth is very small compared to the carrier frequency, the received complex low-pass signal is

$$r(t) = \sum_{n=1}^{N} \alpha_n(t) e^{-j\psi_n(t)} \tag{21}$$

Using (16), the received band pass signal $x(t)$ can be expressed in quadrature form

$$x(t) = r_I(t)\cos 2\pi f_c t - r_Q \sin 2\pi f_c t \tag{22}$$

where

$$r_Q(t) = \sum_{n=1}^{N} \alpha_n(t)\sin\psi_n(t) \tag{23}$$

$$r_I(t) = \sum_{n=1}^{N} \alpha_n(t)\cos\psi_n(t) \tag{24}$$

For large N, the central limit theorem implies that $r_I(t)$ and $r_Q(t)$ can be treated as independent Gaussian processes. Assuming wide-sense stationarity, the autocorrelation of $x(t)$ is

$$\begin{aligned} \varphi_{xx}(\tau) &= E[x(t)x(t+\tau)] \\ &= E[r_I(t)r_I(t+\tau)\cos 2\pi f_c\tau - E[r_Q(t)r_I(t+\tau)\sin 2\pi f_c \\ &= \varphi_{r_I r_I}(\tau)\cos 2\pi f_c\tau - \varphi_{r_Q r_I}(\tau)\sin 2\pi f_c\tau \end{aligned} \tag{25}$$

Note that

$$\varphi_{r_I r_I}(\tau) = \varphi_{r_Q r_Q}(\tau) \tag{26}$$

$$\varphi_{r_I r_Q}(\tau) = -\varphi_{r_Q r_I}(\tau) \tag{27}$$

$\varphi_n(t)$ is independent of $\varphi_m(t)$ for $n=m$, and phases $\varphi_n(t)$ can be assumed to be uniformly distributed over $[-\pi,\pi]$.

$$\begin{aligned}\varphi_{r_I r_I}(\tau) &= E[r_I(t) r_I(t+\tau)] \\ &= \frac{\Omega_p}{2} E[\cos 2\pi f_c \tau] \\ &= \frac{\Omega_p}{2} E[\cos 2\pi f_m \tau \cos\theta]\end{aligned} \tag{28}$$

where $\Omega_p/2 = E[x^2(t)] = E[r_I^2(t)] = E[r_Q^2(t)] = 1/2 \sum_{n=1}^{N} E[\alpha_n^2]$ is the total average received power from all multipath components. Likewise, the cross correlation $\varphi_{r_I r_Q}(\tau)$ is

$$\begin{aligned}\varphi_{r_I r_Q}(\tau) &= E[r_I(t) r_Q(t+\tau)] \\ &= \frac{\Omega_p}{2} E[\sin 2\pi f_m \tau \cos\theta]\end{aligned} \tag{29}$$

For urban areas, it is reasonable to assume that plane waves arrive at the receiver from all directions in the *x-y* plane with equal probability; thus, θ is uniformly distributed over $[-\pi,\pi]$. This isotropic scattering model reduces the expectation in (29) to

$$\begin{aligned}\varphi_{r_I r_I}(\tau) &= \frac{\Omega_p}{2} \frac{1}{2\pi} \int_{-\pi}^{\pi} \cos(2\pi f_m \tau \cos\theta) d\theta \\ &= \frac{\Omega_p}{2} J_o(2\pi f_m \tau)\end{aligned} \tag{30}$$

where $J_o(x)$ is the zero order Bessel function of the first kind. The crosscorrelations become zero for this model.

The power density spectrum (psd) of $r_I(t)$ and $r_Q(t)$ are the Fourier transform of $\varphi_{r_I r_I}(\tau)$ or $\varphi_{r_Q r_Q}(\tau)$,

$$\begin{aligned} S_{r_I r_I}(f) &= F[\varphi_{r_I r_I}(\tau)] \\ &= \frac{\Omega_p}{4\pi f_m} \frac{1}{\sqrt{1-(f/f_m)^2}} \end{aligned} \tag{31}$$

for $|f| \leq f_m$ and zero otherwise.

The autorcorrelation of the received complex low pass signal $r(t)= r_I(t)+jr_Q(t)$ is

$$\begin{aligned} \varphi_{rr}(\tau) &= \frac{1}{2} E\left[r^*(t) r(t+\tau)\right] \\ &= \varphi_{r_I r_I}(\tau) + j\varphi_{r_I r_Q}(\tau) \end{aligned} \tag{32}$$

From (30)

$$\varphi_{xx}(\tau) = \mathrm{Re}[\varphi_{rr}(\tau) e^{j2\pi f_c \tau}] \tag{33}$$

Assuming isotropic scattering, $\varphi_{r_I r_Q}(\tau) = 0$ and $\varphi_{rr}(\tau) = \varphi^*_{rr}(\tau)$, it follows that the power spectral density (psd) of the bandpass waveform $x(t)$ is

$$S_{xx}(f) = \frac{1}{2}[S_{r_I r_I}(f - f_c) + S_{r_I r_I}(-f - f_c)] \tag{34}$$

As $N \to \infty$, the angle of incidence approaches a continuous distribution denoted by $p(\theta)$. The fraction of the total incoming power that arrives between θ and $\theta + d\theta$ is $p(\theta)d\theta$. If the antenna has a gain of $G(\theta)$ at angle θ, then the corresponding received power is $G(\theta)p(\theta)\, d\theta$. The psd of the received signal can be expressed as

$$S_{xx}(f) \mid df \mid = \frac{\Omega_p}{2} \{G(\theta) p(\theta) + G(-\theta) p(-\theta)\} \mid d\theta \mid \tag{35}$$

The frequency of the incident plane wave arriving at angle θ is

$$f = f_m \cos\theta + f_c \tag{36}$$

where $f_m = v/\lambda_c$ is the maximum Doppler shift and, hence

$$\begin{aligned} |df| &= f_m |-\sin\theta d\theta| \\ &= \sqrt{f_m^2 - (f - f_c)^2}\, |d\theta| \end{aligned} \tag{37}$$

Therefore,

$$S_{xx}(f) = \frac{\Omega_p / 2}{\sqrt{f_m^2 - (f - f_c)^2}} \{G(\theta)p(\theta) + G(-\theta)p(-\theta)\} \tag{38}$$

where

$$\theta = \cos^{-1}\left(\frac{f - f_c}{f_m}\right) \tag{39}$$

Here, it is assumed that a vertical monopole antenna is used which reduces $G(\theta) = 3/2$ and with isotropic scattering $p(\theta) = 1/2\pi$, $-\pi \le \theta \le \pi$, so that

$$S_{xx}(f) = \frac{3\Omega_p}{4\pi f_m} \frac{1}{\sqrt{1 - \left(\frac{|f - f_c|}{f_m}\right)^2}} \tag{40}$$

Notice that $S_{xx}(f)$ is limited to the range of frequencies $|f - f_c| \le f_m$.

When the composite received signal consists of a large number of plane waves, the received complex low-pass signal $r(t) = r_I(\tau) + jr_Q(\tau)$ can be modeled as a complex Gaussian random process. In the absence of a LOS component, $r_I(t)$ and $r_Q(t)$ have zero mean. Thus, the received complex envelope $z(t) = |r(t)|$ has a Rayleigh distribution at any time t,

$$p_z(x) = \frac{x}{\sigma^2} \exp\left(\frac{-x^2}{2\sigma^2}\right) \tag{41}$$

For a Rayleigh distributed envelope, the average power of the complex envelope $z(t)$ is $E[z^2]=\Omega_p=2\sigma^2$, the power of the received signal, $r(t)$ is 1/2 $E[z^2]$. This distribution agrees very well with empirical observations for macrocellular applications where no LOS path exists between the transmitter and receiver antennas [7]. The squared envelope $|r(t)|^2$ is exponentially distributed with density

$$p_{z^2}(x) = \frac{1}{2\sigma^2}\exp(\frac{-x}{2\sigma^2}) \tag{42}$$

The autocorrelation of the envelope $|r(t)|$ of a complex Gaussian random process can be expressed in terms of the hypergeometric function $F[.;.;.;.]$ as

$$\begin{aligned}\varphi_{zz}(\tau) &= E[z(t)z(t+\tau)] \\ &= \frac{\pi}{2}|\varphi_{rr}(0)|F\left[-\frac{1}{2};-\frac{1}{2};1,\frac{|\varphi_{rr}(\tau)|^2}{|\varphi_{rr}(0)|^2}\right]\end{aligned} \tag{43}$$

where

$$|\varphi_{rr}(\tau)|^2 = \varphi^2_{r_I r_I}(\tau) + \varphi^2_{r_I r_Q}(\tau)\,. \tag{44}$$

By approximation, the autocorrelation function becomes

$$\varphi_{zz}(\tau) \approx \frac{\pi}{2}|\varphi_{rr}(0)|\left[1+\frac{1}{4}\frac{|\varphi_{rr}(\tau)|^1}{|\varphi_{rr}(0)|^2}\right] \tag{45}$$

The psd of the received envelope can be obtained by taking the Fourier transform of $\varphi_{rr}(\tau)$. The autocovariance function is of interest and is described as

$$\begin{aligned}\mu_{zz}(\tau) &= E[z(t)z(t+\tau)] - E[z(t)]E[z(t+\tau)] \\ &= \frac{\pi}{2}|\varphi_{rr}(0)|\left[1+\frac{1}{4}\frac{|\varphi_{rr}(\tau)|^2}{|\varphi_{rr}(0)|^2}\right] - \frac{\pi}{2}|\varphi_{rr}(0)|\end{aligned} \tag{46}$$

With isotropic scattering $|\varphi_{rr}(\tau)|^2 = \varphi^2_{r_i r_j}(\tau)$ and therefore,

$$\mu_{zz}(\tau) = \frac{\pi\Omega_p}{16} J_o^2(2\pi f_m \tau) \tag{47}$$

On the other hand, assuming isotropic scattering, the Fourier transform of $\mu_{zz}(\tau)$ can be described as

$$\begin{aligned} S_{zz}(f) &= \frac{\pi}{8|\varphi_{rr}(0)|} \int_{-f_m}^{f_m - |f|} S_{rr}(x) S_{rr}(x + |f|) dx \\ &= \frac{\Omega_p}{64\pi} \frac{1}{f_m} K\left(\sqrt{1 - \left(\frac{f}{2f_m}\right)^2}\right) \end{aligned} \tag{48}$$

for $0 \leq |f| \leq 2f_m$, where $K(.)$ is the complete elliptic integral of the first kind, defined by

$$K(\gamma) = \int_0^1 \frac{dx}{\sqrt{(1-x^2)(1-(1-\gamma^2)x^2)}} \tag{49}$$

Note that $S_{zz}(f)$ is always real, positive and even. It is centered about f=0 with a spectral width of $4f_m$, where f_m is the maximum Doppler frequency.

For the flat fading channel model, the inverse signal bandwidth is much greater than the time spread of the propagation path delays. Under this condition, all frequencies in the transmitted signal will experience the same random attenuation and phase shift due to multipath fading.

If the range in the propagation path delays is large compared to the inverse signal bandwidth, the frequency components in the transmitted signals will experience different phase shifts along the different paths. As differential path delays

become large, even closely separated frequencies in the transmitted signal can experience significantly different phase shifts. Amplitude and phase distortion are introduced into the signal by the channel and such a channel is said to exhibit frequency selective fading.

Multipath fading channels can be modeled as time variant linear filters, whose inputs and outputs can be described in both the time and frequency domains. The complex low pass impulse response relates the complex low-pass input and output time waveforms, $u(t)$ and $r(t)$ through convolution.

$$r(t) = \int_{-\infty}^{\infty} u(t-\tau)c(t,\tau)d\tau \tag{50}$$

The low pass impulse response $c(t,\tau)$ is the input delay spread function. It can be interpreted as the channel response at time t due to an impulse applied at time t-τ. In the frequency domain the input and output spectra, $U(f)$ and $R(f)$, are related through the output Doppler spread function $H(f\text{-}v,v)$.

$$R(f) = \int_{-\infty}^{\infty} U(f-v)H(f-v,v)dv \tag{51}$$

The frequency shift variable v can be interpreted as the Doppler shift that is introduced by the channel and the function shows the effect of Doppler shift to the output spectrum.

Let $\Omega_v=E[z(t)]$ denote a slow variation of the fading envelope. This mean represents the envelope level averaged over a distance of a few wavelengths. Field studies have shown that the mean follows a log normal distribution [7]. It is also

known that log normally distributed variable can be described as a Gaussian distributed variable in units of dB, thus the pdf of the mean is

$$p(\Omega_{v(dB)}) = \frac{1}{\sqrt{2\pi}\sigma_{\Omega}} \exp\left\{ -\frac{\left(\Omega_{v(dB)} - \mu_{\Omega_v}\right)^2}{2\sigma_{\Omega}^2} \right\} \tag{52}$$

Path loss predicts how the mean signal power decays with distance from a base station. It is known that the received signal power decays with the square of the path length in free space and follows other more complicated laws in different mobile radio environments.

3.2 Simulation of channel models

System performance over all ranges of expected channel variation makes control of the channel fading and Doppler shifts essential. For this reason, channel simulators that are derived from theoretical principles are of interest. Here two methods are introduced.

3.2.1 Jakes' method

As described in the previous subsection, the received complex low pass signal can be described as a summation of plane waves. Assuming stationarity ($f_{D,n}(t)=f_{D,n}$) and equal strength multipath components ($\alpha_n=1$),

$$r(t) = \sum e^{-j(\hat{\varphi} + 2\pi f_m t \cos\theta_n)} \tag{53}$$

$$\hat{\varphi}_n = 2\pi\left(f_c + f_m\right)\tau_n \tag{54}$$

$$\theta_n = \frac{2\pi n}{N} \tag{55}$$

where n ranges from 0 to N, assuming N is a sufficiently large number. Assuming $N/2$ is an odd integer, the series can be rearranged to give:

$$r(t) = \left\{ \sum_{n=1}^{N/2-1} \left[e^{j(2\pi f_m t \cos\theta_n + \hat{\varphi}_n)} + e^{-j(2\pi f_m t \cos\theta_n + \hat{\varphi}_{-n})} \right] + e^{j(2\pi f_m t + \hat{\varphi}_N)} + e^{-j(2\pi f_m t + \hat{\varphi}_{-N})} \right\} \tag{56}$$

It can be seen in (54) that there are terms in the summation where their frequencies overlap. Without any loss of generality, it will be convenient to represent the signal in terms of waves whose frequencies are unique. This reduces the summation to:

$$r(t) = \left\{ \sqrt{2} \sum_{n=1}^{\frac{1}{2}\left(\frac{N}{2}-1\right)} \left[e^{j(2\pi f_m t \cos\theta_n + \hat{\varphi}_n)} + e^{-j(2\pi f_m t \cos\theta_n + \hat{\varphi}_{-n})} \right] + e^{j(2\pi f_m t + \hat{\varphi}_N)} + e^{-j(2\pi f_m t + \hat{\varphi}_{-N})} \right\} \tag{57}$$

If N is large enough, by the central limit theorem, the summation is approximately a complex Gaussian process, so a Rayleigh distribution function is generated.

It has been suggested that the summation of six simplified sinusoids is sufficient to generate Rayleigh coefficients [25]. It has also been mathematically proven that the autocorrelation of the electric field suggested by this model does indeed closely approximate the required autocorrelation function when $N \leq 34$ [25]. This model is realized by the in-phase and the quadrature components as shown in Figure 3-1, where:

$$\begin{aligned} r(t) &= r_I(t) + j r_Q(t) \\ r_I(t) &= \frac{4\sigma^2}{\sqrt{N}} \left(\sqrt{2} \sum_{n=1}^{N_a} \cos\beta_n \cos 2\pi f_n t + \cos\alpha \cos 2\pi f_m t \right) \\ r_Q(t) &= \frac{4\sigma^2}{\sqrt{N}} \left(\sqrt{2} \sum_{n=1}^{N_a} \sin\beta_n \cos 2\pi f_n t + \sin\alpha \cos 2\pi f_m t \right) \end{aligned} \tag{58}$$

where

$$2\pi f_m t \cos\alpha_n = 2\pi f_n t \tag{59}$$

$$\varphi_n = \beta_n = \frac{\pi n}{N_o}$$

$$\varphi_N = \alpha$$

In many cases, it is desirable to generate multiple uncorrelated faded carriers. Jakes' method may be extended up to M fading signals by using the same low frequency oscillators. This is accomplished by giving the n^{th} oscillator the additional phase shift $\gamma_{nj} + \beta_n$, for $1<j<M$, where

$$\beta_n = \frac{\pi n}{M} \tag{60}$$

$$\gamma_{nj} = \frac{2\pi (j-1)n}{M} \tag{61}$$

for $1 \leq n \leq M$.

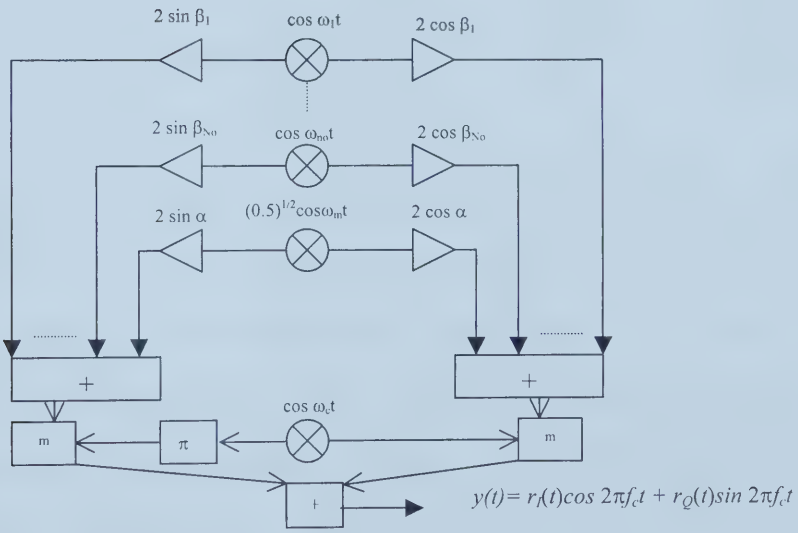


Figure 3-2: Jakes' channel model.

The cross correlations between the different faded envelopes can be computed as

$$\varphi_{r_i r_j}(\tau) = \frac{E[r_i^*(t) r_j(t+\tau)]}{\sqrt{E[|r(t)|^2]E[|r_j(t)|^2]}} \tag{62}$$

[7] showed that the cross correlations simulated by Jakes' model are not always close to zero. The significant cross correlation between the different faded envelopes that are generated is undesirable. Also, [26] has shown that Jakes' model may violate stationary properties of a Rayleigh channel model. Thus, the IFFT based simulator is preferred.

3.2.2 IFFT Based Simulator Design

The previous section describes a Rayleigh channel model that is generated in the time domain. In this section, the model as described in [27] is produced in the frequency domain. Define $A(k)$ and $B(k)$ as zero mean, independent Gaussian variables, where

$$\begin{aligned} \overline{A(k)} &= 0 = \overline{B(k)} \\ \overline{A^2(k)} &= \sigma^2 = \overline{B^2(k)} \end{aligned} \tag{63}$$

These variables will be combined to form a complex variable, $X(k)$, and will be time translated to form the in-phase and quadrature components of the electric field. However, in order to incorporate the required Doppler spread into the channel model, the Gaussian random processes have to be shaped to the desired spectrum. In the frequency domain, this means multiplying the power spectral density of the white Gaussian by the filter transfer function which is defined by the maximum Doppler shift, $F(k)$. These two components are combined to form a complex variable, $X(k)$,

$$X(k) = F(k)A(k) - jF(k)B(k) \tag{64}$$

$X(k)$ is translated into time domain, $x(n)$, where

$$
\begin{aligned}
x(n) &= \frac{1}{N}\sum_{k=0}^{N-1}[F(k)A(k) - jF(k)B(k)]e^{j\frac{2\pi kn}{N}} \\
&= \frac{1}{N}\left[\left(\sum_{k=0}^{N-1}[F(k)A(k)\cos\frac{2\pi kn}{N} + F(k)B(k)\sin\frac{2\pi kn}{N}\right)\right. \\
&\quad \left. + j\left(\sum_{k=0}^{N-1}[F(k)A(k)\sin\frac{2\pi kn}{N} - F(k)B(k)\cos\frac{2\pi kn}{N}\right)\right] \\
&= \frac{1}{N}\left[x_R(n) + jx_I(n)\right]
\end{aligned}
\tag{65}
$$

The averages of the $x_R(n)$ and $x_I(n)$ sequences are zero because the summation of sinusoids will be zero. The autocorrelation of $x_R(n)$, $\varphi_{x_r x_r}(\tau)$, is :

$$
\begin{aligned}
\varphi_{x_r x_r}(\tau) &= \frac{1}{N^2}\sum_{k=0}^{N-1}\left[\begin{array}{l} F^2(k)A^2(k)\cos\frac{2\pi kn}{N}\cos\frac{2\pi k(n-\tau)}{N} + \\ F^2(k)A(k)B(k)\cos\frac{2\pi kn}{N}\sin\frac{2\pi k(n-\tau)}{N} + \\ F^2(k)A(k)B(k)\sin\frac{2\pi kn}{N}\cos\frac{2\pi k(n-\tau)}{N} + \\ F^2(k)B^2(k)\sin\frac{2\pi kn}{N}\sin\frac{2\pi k(n-\tau)}{N} \end{array}\right] \\
&= \frac{\sigma^2}{N^2}\sum_{k=0}^{N-1}F^2(k)(\cos\frac{2\pi kn}{N}\cos\frac{2\pi k(n-\tau)}{N} + \sin\frac{2\pi kn}{N}\sin\frac{2\pi k(n-\tau)}{N}) \\
&= \frac{\sigma^2}{N^2}\sum_{k=0}^{N-1}F^2(k)\cos\frac{2\pi k(\tau)}{N} \\
&= \frac{\sigma^2}{N^2}\sum_{k=0}^{N-1}F^2(k)(\frac{e^{i\frac{2\pi k\tau}{N}} + e^{-i\frac{2\pi k\tau}{N}}}{2}) \\
&= \frac{\sigma^2}{2N}\left[\left(\frac{1}{N}\sum_{k=0}^{N-1}F^2(k)e^{i\frac{2\pi k\tau}{N}}\right) + \left(\frac{1}{N}\sum_{k=0}^{N-1}F^2(k)e^{i\frac{2\pi k\tau}{N}}\right)^*\right] \\
&= \frac{\sigma^2}{2N}\left[g(\tau) + g(\tau)^*\right] \\
&= \frac{\sigma^2}{N}\mathrm{Re}\left[g(\tau)\right] \\
&= \varphi_{x_i x_i}(\tau)
\end{aligned}
\tag{66}
$$

Similarly, it can be shown that the cross correlation, $\varphi_{x_r x_i}(\tau)$, is

$$\varphi_{x_r x_i}(\tau) = \frac{\sigma^2}{N} \mathrm{Im}[g(\tau)] \tag{67}$$

Since $x_R(n)$ and $x_I(n)$ are not correlated, $\varphi_{x_r x_i}(\tau)$ must be zero, which implies that the imaginary part of $g(\tau)$ has to be zero; thus, $g(\tau)$ is a real number. In order to generate $g(\tau)$ with the above requirement, the corresponding frequency domain component of $g(\tau)$, $F^2(k)$, has to satisfy the condition that $F^2(k) = F^2(N\text{-}k)$. It is known that the autocorrelation function, $\varphi_{x_r x_r}(\tau)$, can be translated into a frequency domain variable, $F^2(k)$ [7]. This variable is better known as the power spectral density, $S(f)$. Therefore, $F(k)$ is set to equal the square root of the power spectral density, $(S(f))^{1/2}$. Also, as shown in equation (64), at τ=0:

$$\varphi_{x_r x_r}(0) = \sigma^2 = \frac{\sigma^2}{N} g(0) \tag{68}$$

and

$$g(0) = \frac{1}{N} \sum_{n=0}^{N-1} S(kf) e^{j\frac{2\pi kn}{N}} = \frac{1}{N} \sum_{n=0}^{N-1} S(kf) \tag{69}$$

The Rayleigh coefficients generated have to be adjusted to obtain the required autocorrelation magnitude. Therefore, the magnitude of $x(n)$ will have the desired Rayleigh coefficients with the required statistics.

To implement an IFFT-based simulator, the following steps are used:

1) Specify the number of frequency domain points (N) required and the maximum Doppler shift (f_m).
2) Compute $F(k)$ where Δf=2 f_m/ (N-1). As mentioned above, $F(k) = (S(k\Delta f))^{1/2}$ where $S(f)$ is as specified in (39).

3) Generate N zero mean Gaussian random variables $A(k)$ and $B(k)$ and combine the components as suggested in (63) to form $X(k)$.

4) Perform an IFFT on the resulting frequency domain signals and take the magnitude of the result to obtain N point time series of a simulated Rayleigh fading signal with the proper Doppler spread and time correlation.

3.3 Power control

The performance of wireless radio systems is usually limited by interference rather than noise, and therefore the probability of co-channel interference is of primary concern. The effect of co-channel interference on the radio link performance depends on the ability of the radio receiver to reject co-channel interference. To minimize the co-channel interference, several techniques are proposed: frequency reuse patterns, which ensure that the same frequencies are not used in adjacent cells; efficient power control, which minimizes the transmitter power; co-channel interference cancellation techniques, and orthogonal signaling. This section provides a short overview of power control.

In a CDMA system, users are separated by codes and the allocated frequency may be shared by all users in all cells. The capacity of the system is limited by interference since users are neither separated in time or frequency. As a result, co-channel interference is inherently strong. A single user exceeding the limit on transmitted power could inhibit the communication of all other users. Therefore, power control is vital for system operation on the reverse link. On the forward link, operation without power control is possible but inefficient.

Power control is used to maintain a minimal transmitted level at the base station on a forward link to keep the co-channel interference low. In the reverse link, it maintains the minimum necessary transmitted power for reliable communication. Forward and reverse links do not usually share the same frequency, and therefore the fading is significantly different. However, the large scale channel fluctuations due to shadowing and path loss are basically the same.

Power control systems are designed to compensate not only for signal strength variations due to the varying distance between base station and mobile, but must also attempt to compensate for signal strength fluctuations typical of a wireless channel. There are two types of power control methods. One is slow transmit power control (TPC) and the other is fast TPC. Slow TPC sets the transmitted power level inversely proportional to its estimate of the channel attenuation. This enables the average power of the received signal to be held relatively constant. Since the forward link and reverse link are uncorrelated, it can only compensate for propagation and shadowing losses. In fast TPC, the fading fluctuations in both links are estimated and the transmitted power is adjusted accordingly.

If the base station receiver uses diversity combining, the fading on each branch will be independently Rayleigh distributed with a specific Doppler spread. It has been shown that the resulting channel attenuation after fast TPC can be approximated by a log-normal distribution with a standard deviation of σ_S (dB),

where the magnitude of σ_S is proportional to $f_d T_p$ and f_d is the maximum Doppler shift of the system [28].

It is shown in [24] that as the Doppler shift increases, the tracking ability of fast TPC decreases, and as a result, the standard deviation of the TPC error, σ_S, increases. Decreasing the data rate has the same effect since the number of data points available to estimate the same spectrum is less. Although a larger control step size is desirable to track the rapid change of the signal level, it causes larger TPC error in low Doppler shifts, so a step size of 1 dB is suggested [24]. At a Doppler shift of 5.6 Hz, the standard deviation of the TPC error is found to be about 1.7-2.0 dB for both the Vehicular B channel defined in [29] and the 2-path Rayleigh channel [28].

3.4 Simulation Structure

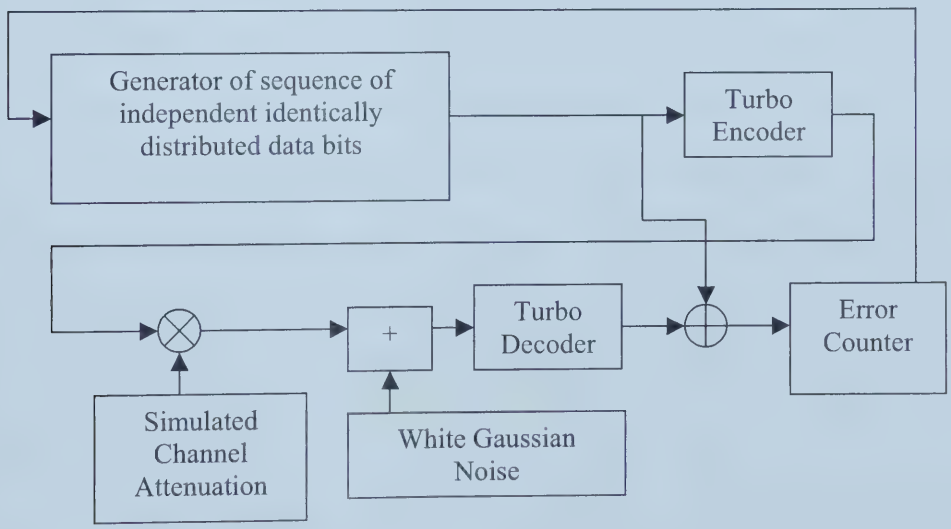


Figure 3-3: Simulation Structure.

Figure 3-4 is a block diagram of the structure used for the simulation. A random number generator is used to generate a block of information bit sequence. If the generated number is less than 0, it will be encoded as 0 otherwise it is encoded as 1. The information sequence is generated block by block where the block size is dependent on the interleaver size considered.

The block of information sequence is encoded by the turbo encoder as described in Figure 2-1. The resulting multiplexed sequence is then multiplied by the attenuation of the channel model. For the purpose of simulation in this thesis, parameters are simplified when modelling the channel attenuation.

A model that simulates data rates of 64 kb/s with 0.625 ms/slot and a Doppler shift of 5.55 Hz is generated. Assuming 2-path Rayleigh channel fading, two spatially separated antennas are used to receive the transmitted signal. There are two fingers on each antenna resulting in four independent Rayleigh fading channels of equal gain. Hence, four independent IFFT-simulated Rayleigh fading channels are modeled. 1024 frequency points were generated for each Rayleigh fading channel for this simulation. The output signals are maximal ratio combined in this one-cell based channel model. Shadowing and path loss are not simulated in this system as it is assumed to be mitigated through power control.

SIR measurements were taken at the output of the maximal ratio combined symbol [24]. All 40 symbols within the slot are used for the SIR measurement. The measured SIR is as follows:

$$S = \left(\frac{1}{N} \sum_{i=1}^{N} r_i \hat{r}_i^* \right)^2 \qquad (70)$$

$$I = \frac{1}{N} \sum_{i=1}^{N} \left(r_i \hat{r}_i^* \right)^2 - S$$

$$SIR = \frac{S}{I}$$

where r_i is the i^{th} demodulated signal after rake combining and subscript i means the i^{th} symbol in the slot, $\hat{r}_i$ is the result of the decision of symbol r_i , S is the received signal power, I is the average interference plus background noise power, and N is the number of symbols used in each slot for SIR measurement. The measurement of I is obtained from a sum of weighted past measurements of I, thus providing a more accurate measurement:

$$\bar{I}(k) = \alpha \bar{I}(k-1) + (1-\alpha)\bar{I}(k) \qquad (71)$$

The value α=0.999, as suggested in [30], is used in this model

After obtaining 40 symbols' measurement, the resulting SIR is compared with the target SIR. If the measured SIR is less than the target SIR, the transmitted signal gain is increased by 1 dB and vice versa.

In this model, the received signal is found to be log-normally distributed with a standard deviation of approximately 0.8 dB, as shown in Figure 3-3. This small standard deviation indicates remarkable tracking capability, and may be a result of perfect channel estimation, errorless TPC command transmission or no TPC control delay. To simplify the channel attenuation, a log-normally distributed random variable is used to model the channel attenuation. For the sake of worst case analysis, the standard deviation for the log-normal fading used in the simulations was fixed at 2.0 dB.

The attenuated sequence is then added with white Gaussian noise. The variance of the noise is determined by the signal-to-noise ratio that is being considered in the simulation.

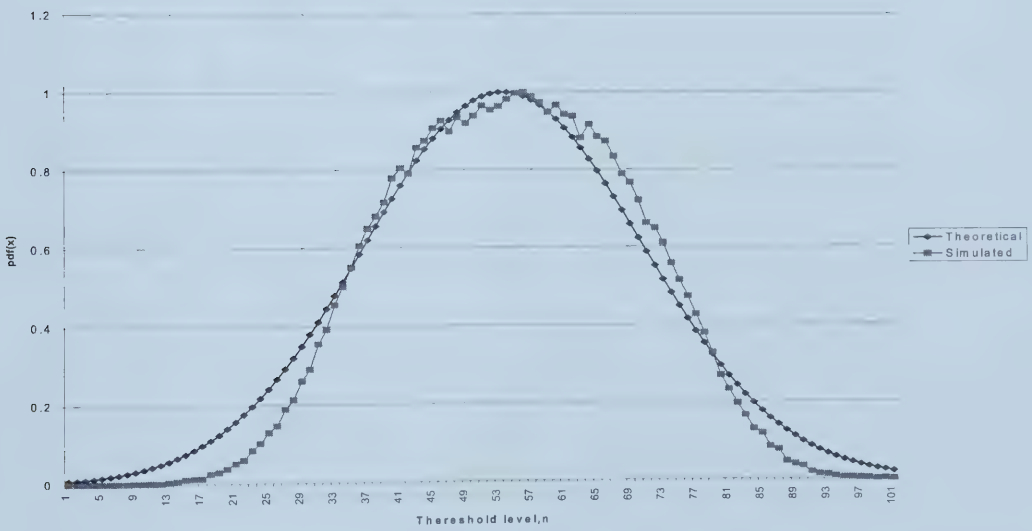


Figure 3-4: Probability density function (pdf) of power controlled signal.

The received sequence is decoded by the turbo codes decoder described in Figure 2-2. The output of the decoder and the input sequence is added using modulo-2 arithmetic. If there is a decoding error, the result of the operation is 1. If the sum of the errors is less than 100, another block of information bits is generated. This process continues until at least 100 errors have been collected.

IV. SIMULATION RESULTS

The basic structure of turbo codes was established in Chapter II. Parameters of interest such as the constituent codes, interleaver, decoding iterations, and trellis termination have also been introduced briefly. The effects of these parameters, however, are not well understood. The focus of this thesis is to consider these parameters of interest of turbo codes in the context of IMT-2000 system requirements. Investigation of these parameters is presented in this chapter.

This chapter begins with the selection of the interleaver with respect to its size and type. Decoding iterations and constituent codes of turbo codes are investigated as well. Following a discussion of constraint length, error events and constituent generator polynomials of the constituent codes, the effect of dual termination is considered.

4.1 Selection of interleaver

4.1.1 Interleaver Size

In general, it is known that the performance of turbo codes improves with increasing interleaver size; however, interleaver size translates into decoding delay. In the current IMT-DS proposal, the frame length is 10 ms with a data rate of 64 kb/s. With this configuration, the standard size for a block will be 640 bits long. Since a shorter size translates to shorter delay, shorter interleaver sizes are considered.

Here, systems with interleavers of size 160, 320 and 640 were simulated. Recursive systematic convolutional codes of constraint length K=4 and polynomials

(13,17) were used as the constituent codes. S-random interleavers were considered because the prime interleaver is not defined for interleavers as short as 160. Decoded bit error rates are plotted versus signal-to-noise ratio, E_b/N_o, in Figure 4-1.

As expected, performance significantly improved with increased frame size. This is a result of the increased randomness of the interleaving process and, in particular, a higher probability that low input weight sequences are permuted to generate encoded sequences of higher weight.

As depicted in Figure 2-2, decoding includes interleaving and deinterleaving, therefore, an increase in frame size increases decoding complexity and delay. However, with increasing speeds of ASICs and recent simplifications to the constituent decoding algorithms [17,20], the increase in processing time for iterative decoding due to the increase in frame size may not be significant.

For voice services, minimal delay is required but bit error rates of up to 10^{-3} are acceptable [31]. In this case, relatively short frames may prove the best alternative. Further analysis of short frame transmission will be discussed in the next section. Data services, on the other hand, are usually able to tolerate significant delay but typically have a requirement for higher reliability. For these services, longer frames may be required.

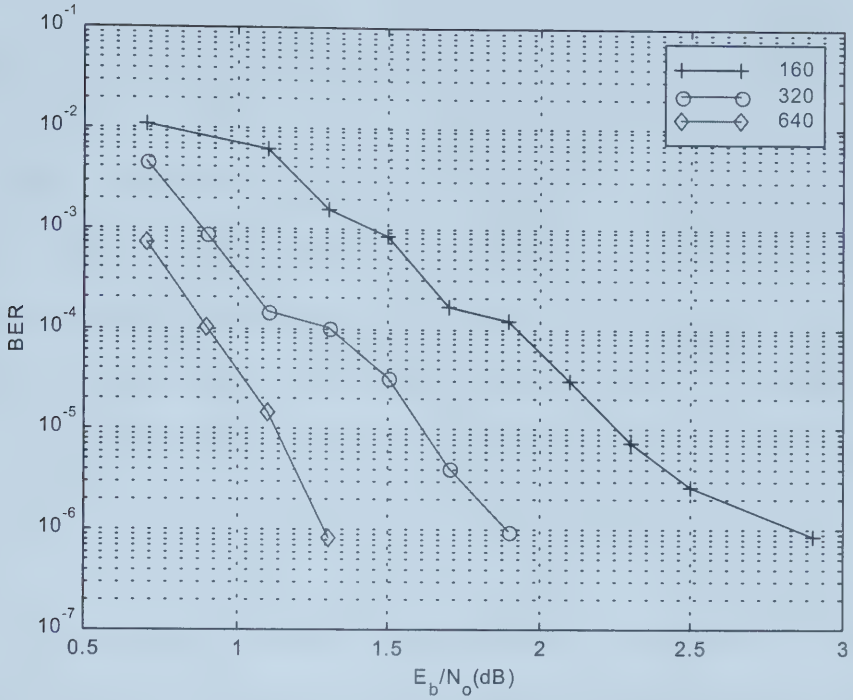


Figure 4-1: Turbo code performance with different sized interleavers. S-random interleaver, *K*=4 (13,17) RSC constituent codes, maximum 8 decoding iterations.

4.1.2 Interleaver Type

Interleavers impact the performance of turbo decoding through their randomness and by permuting the information sequence so as to map critical sequences to non-critical sequences or result in sequences with higher output weight. Prime and S-random interleaver structures are expected to permute critical sequences such that the unpermuted and permuted sequences are independent. This section is focused on these interleavers.

Figures 4-2(a) and (b) depict simulation results for spread random and prime interleavers for constraint length 4 and 5 constituent codes with a maximum of 4 or 8

decoding iterations and a frame size of 640 information symbols. Constituent codes are specified in octal form unless specified.

Since the spread random interleaver is inherently a more random interleaver than the algorithmic prime interleaver, it is expected that the performance of the spread interleaver will be better than the performance of the prime interleaver. Figure 4-2 did not confirm this expectation. Both interleavers have similar performance. It can also be concluded that the difference in performance between 4 and 8 decoding iterations is more uniformly pronounced for the prime interleaver.

To explain these unexpected simulation results, the performance of turbo decoders for these configurations can be bounded using a union bound. Although this bound is valid only for an additive white Gaussian noise (AWGN) channel and tight only for high E_b/N_o[2], it gives insight into the reason for the performance of turbo codes using these interleavers.

As seen in Chapter II, for encoded sequences that differ by the Hamming distance *d*, an upper bound on the error probability of the error correcting code is given by:

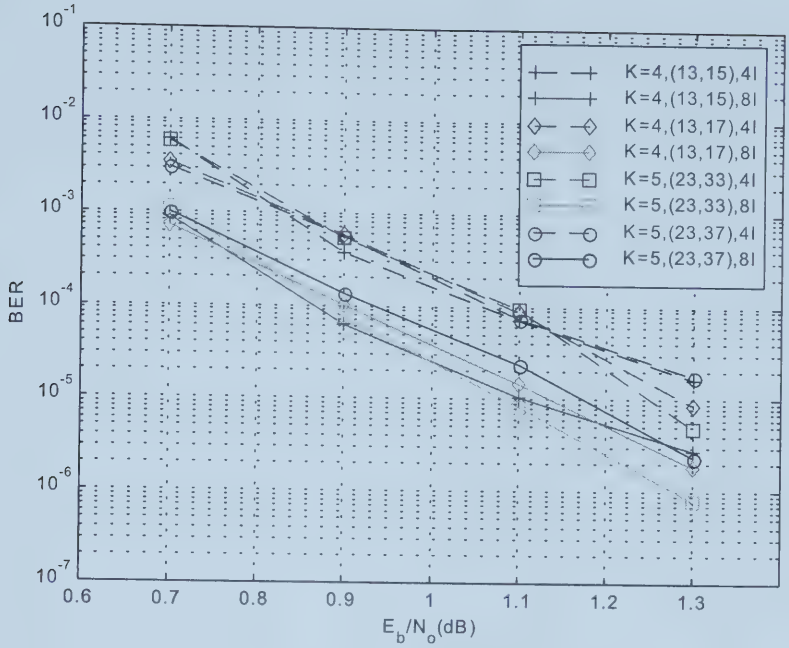
$$BER \leq \sum_{d_{free}}^{\infty} a_d f(d) Q\left(\sqrt{R_c \frac{E_b}{N_o} d}\right) \tag{72}$$

where $Q(\cdot)$ denotes evaluation of the area under the tail of the Gaussian probability distribution function and is dependent of the signal-to-noise ratio, E_b/N_o, and the

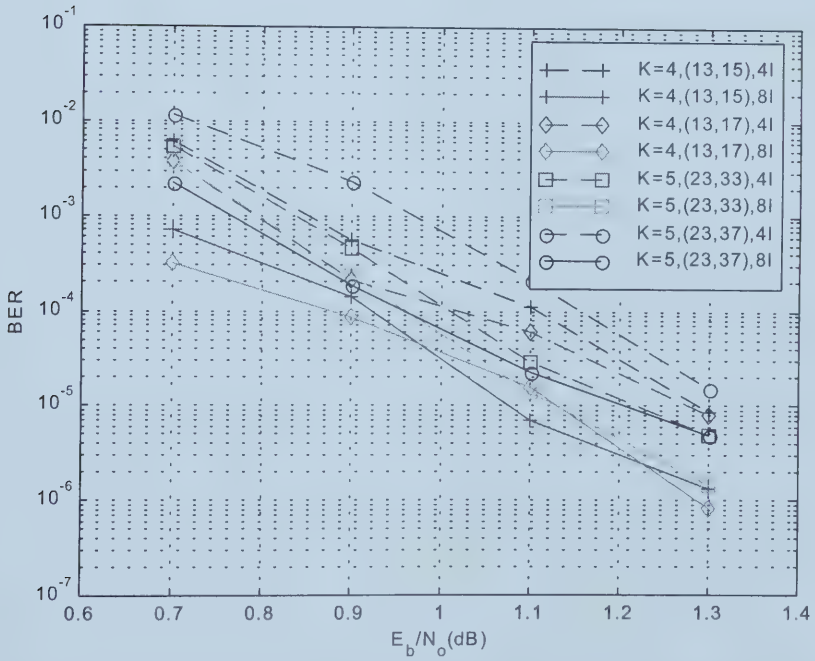
code rate, R_c, of the system [2], a_d is the number of encoded sequences of weight d in the code, and $f(d)$ denotes the weights of the input sequences that generate these sequences. These values can be obtained from the transfer function of the code or from brute force evaluation of code sequence statistics.

Since convolutional codes are linear codes, the decoding error probability is dominated by sequences with low weight [11]. Evaluation of this bound is limited to weight-2 and weight-3 data sequences. In this bound evaluation, the evaluated interleavers permute input sequences of weight-two and three before entering the second encoder. The weight of encoded sequences in the evaluation of equation (72) is the multiplexed output of the encoders and the information sequence. Performance bounds for these codes whose simulated performance were depicted in Figures 4-2(a) and (b) are given in Figures 4-3(a) and (b). Assuming input sequences of higher weight are insignificant in the bound calculation, only input sequences of weight–two and three are used in this performance bound calculation.

Counter to the simulated results shown in Figure 4-2(a), Figure 4-3(a) indicates that turbo codes using constituent codes with polynomials (13,15) exhibit the best performance from use of the prime interleaver. Also, this figure indicates that the prime interleaver must be better than the spread interleaver in mapping critical sequence to non-critical sequences in these codes.

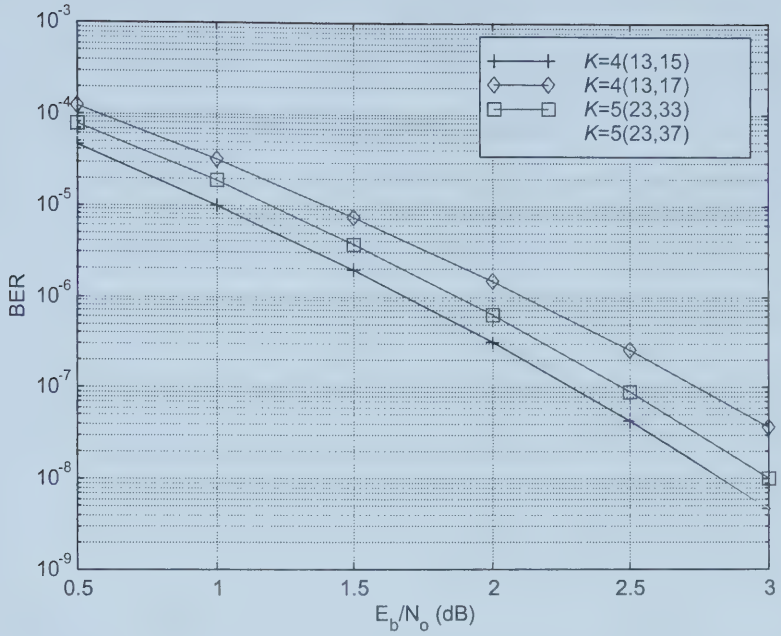


(a) Prime interleaver

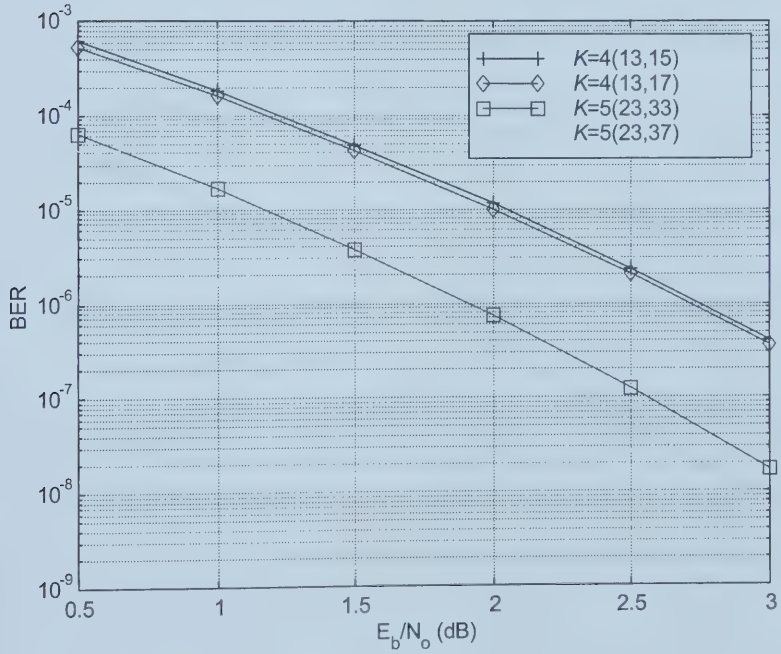


(b) S-random interleaver, S_1=S_2=17

Figure 4-2: Turbo code performance with prime and S-random-random interleavers of size 640; various RSC constituent codes, maximum 4 or 8 iterations.



(a) Prime Interleaver



(b) S-random interleaver, S_1=S_2=17

Figure 4-3: Truncated union bound with prime and S-random interleavers of size 640; various constituent codes.

Performance results for the spread interleaver, on the other hand, confirmed performance expectations given the search for codes reported in [9]. For turbo codes of constraint length 4, polynomials (13,17) were suggested and for constraint length of 5, polynomial (23,37) were suggested in [9]. Regardless of the type of interleaver used in the bound calculation, (23,37) is suggested in [9] to have superior performance because of its distance and multiplicity properties. However, the simulation results given in Figure 4-2(a) and (b) indicate otherwise. This will be investigated in the Section 4.3.3.

4.2 Number of decoding iterations

Iterative decoding is a unique feature of turbo codes that helps achieve excellent performance. It is generally accepted that turbo decoding performance improves with the number of iterations performed, but further improvements become negligible after several iterations.

Consider the turbo codes whose simulated performance was reported in Figure 4.1. With a frame size of 160, to reach a BER of 10^{-3}, an E_b/N_o of 1.5 dB was required. However, by doubling the size of the frame, the E_b/N_o required for the same BER dropped to about 0.7 dB. Note that these simulation results were for a maximum of eight decoding iterations. If the performance of the short turbo codes can instead be improved by increasing the number of decoding iterations, perhaps it would be a better alternative than increasing the frame size.

Increasing the number of decoding iterations would also increase the delay, but the required delay before the decoding process can begin is halved with this smaller frame size. Since the decoding process starts after the whole block arrives, this delay is also dependant on the line rate. The iterative processing delay, on the other hand, is dependant on the speed of the processor or the implementation of turbo codes.

Figure 4-4 shows the impact of decoding iterations on constituent codes with a constraint length of K=4 with constituent code polynomials (13,17) and a frame size of 160 when the spread random interleaver is used.

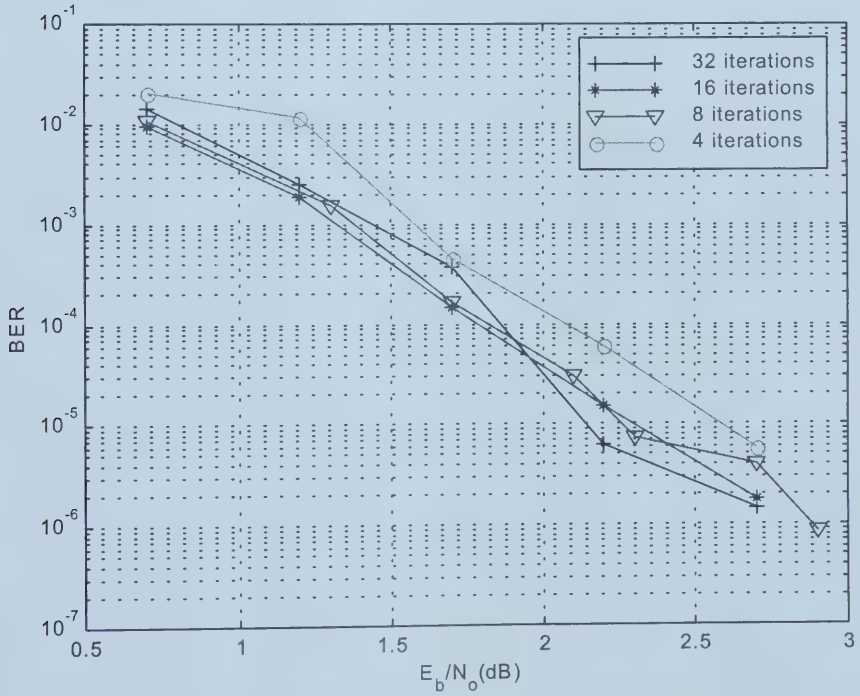


Figure 4-4: Turbo decoding performance for various maximum number of decoding iterations for S-random interleaver of size 160, S_1=S_2=5, K=4 (13,17) constituent codes.

In this simulation, the codes will stop decoding once there is no error within the frame. It is clear from this figure that there is no significant improvement in

performance as the maximum number of iterations is increased from 8 to 16 and 32. This may be explained by the fact that the small interleaver limits the randomness of the permutation and the ability to map critical sequences to non-critical sequences. In some cases, more decoding iterations may lead to the accumulation of numerical errors. The number of errors within this single errored frame may have countered the greater accuracy of other frames.

Similar simulation is considered with a prime interleaver of size 640 in Figure 4-5. The results confirm that an increase in the number of decoding iterations helps

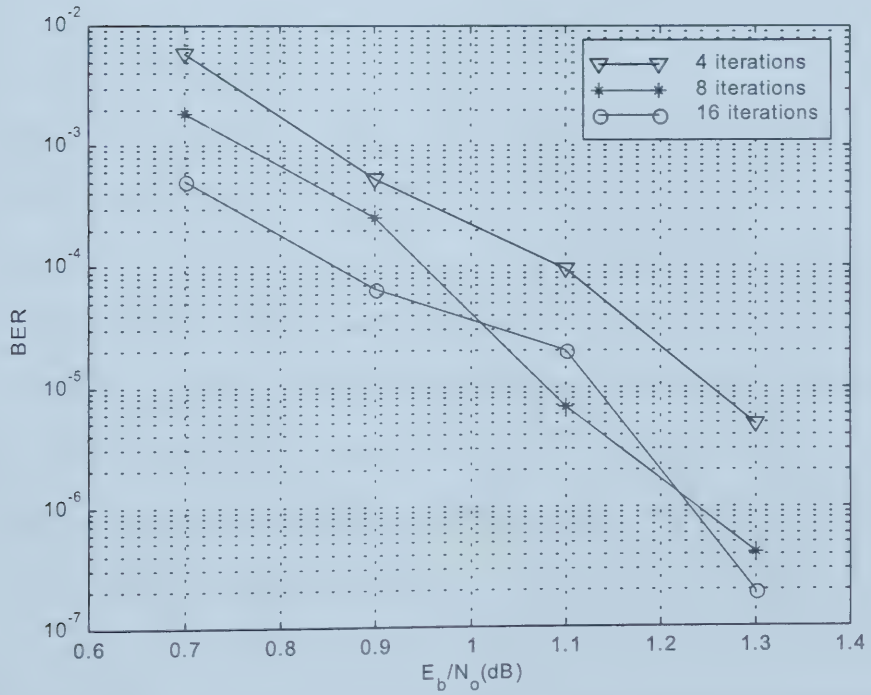


Figure 4-5: Turbo decoding performance for various maximum number of decoding iterations for prime interleaver of size 640, *K*=5 (23,33) constituent codes.

improve the performance. The interleaver is large enough to maintain some degree of independence in the extrinsic information even after 8 iterations, enabling decoding performance to continue to improve as the number of decoding iterations increases. At high E_b/N_o, however, this improvement is not evident. This indicates that as E_b/N_o increases, more iterations are ineffective in correcting the few error sequences that remain incorrectly decoded after 8 iterations. This suggests that with few critical sequence errors, improvement is marginal.

4.3 Constituent code polynomials

4.3.1 Constraint Length

It is well known that with convolutional codes, decoding performance improves with an increase in d_{free}. In general, d_{free} increases with increasing constraint length [2]. For that reason, it might be expected that the performance of turbo codes will improve as the constraint length of the constituent codes increases, however, simulation results indicate otherwise.

Figure 4-6 shows BER results for codes of constraint length ranging from 3 to 6, a frame size of 640, use of a prime interleaver, and a maximum of eight decoding iterations. Counter to expectations, turbo codes with constituent codes of constraint length 6 performed worse than codes with constituent code of constraint lengths 4 and 5. Also, the performance with K=5 only slightly exceeded the performance with K=4, and only at high signal-to-noise ratios. The code with K=3 was the worst overall.

Clearly, these simulation results do not agree with expectations based solely on constraint length. Codes of longer constraint length did not result in the best performance as anticipated. One reason for this unexpected result lies in the ability of the prime interleaver to permute critical sequences to non-critical sequences for different constraint length constituent codes. Recall that the distance separating logic ones in critical sequences depends on the feedback polynomial. All feedback polynomials in the codes whose performance is depicted in Figure 4-6 are primitive, therefore their critical sequences consist of two logic ones separated by nP-1 logic zeros, where $P=2^{K-1}$. Critical sequences corresponding to low values of n are of greatest concern because they are the dominating factor in equation (72).

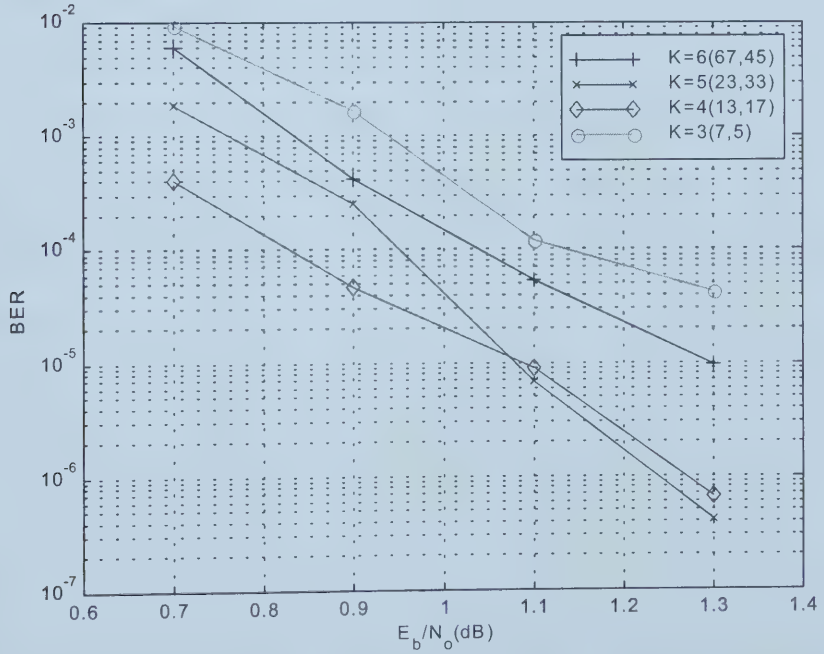
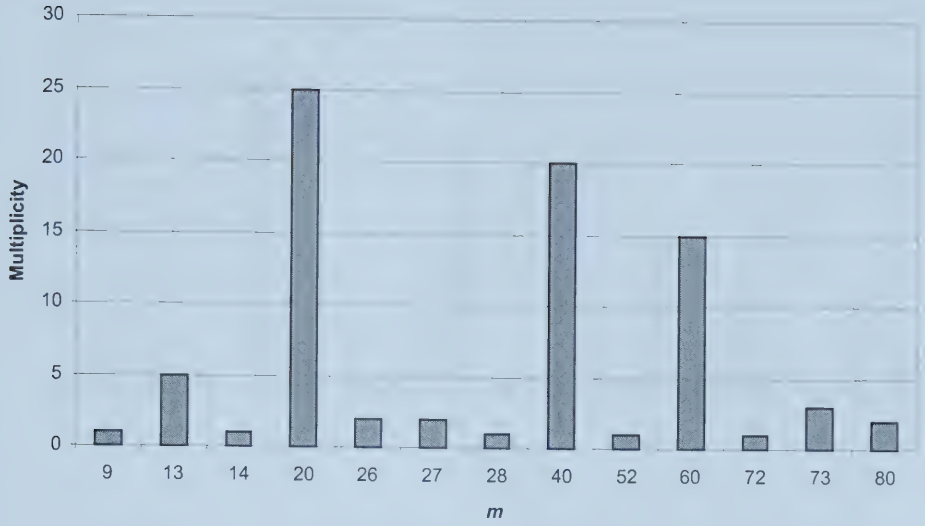


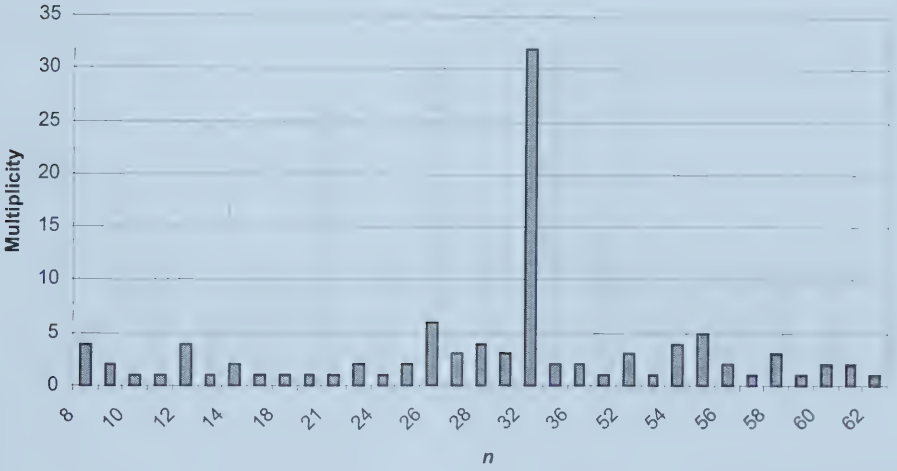
Figure 4-6: Turbo code performance with constituent codes of varying constraint length. Prime interleaver of size 640, maximum of 8 decoding iterations.

Taking into account the logic ones that start and end the critical sequences, the overall length of critical sequences is $nP+1$. The length of critical sequences before permutation is denote as $nP+1$ and of critical sequences after permutation as $mP+1$. When $n=m$, the interleaver has failed to break the critical sequence; at low values of n or m, low weight output sequences are generated. Here the interest rests on the length where n or m equals to 1 for constituent codes with $K=4$ through 6, respectively.

For $K=4$ constituent codes, Figure 4-7 shows that the prime interleaver maps critical sequences of length 8 to critical sequences of at least 9x7+1 = 64. Figure 4-8 shows that for constituent codes of constraint length $K=5$, there are two critical sequences of length 16 that are mapped to critical sequences of length 31 and several that are mapped to length 61. Figure 4-9 shows that for constituent codes of constraint length $K=6$, there are six critical sequences of length 32 that are permuted to critical sequences of this same length. This is indicative of a relatively high probability of decoding failure as confirmed by the simulation results presented in Figure 4-6.

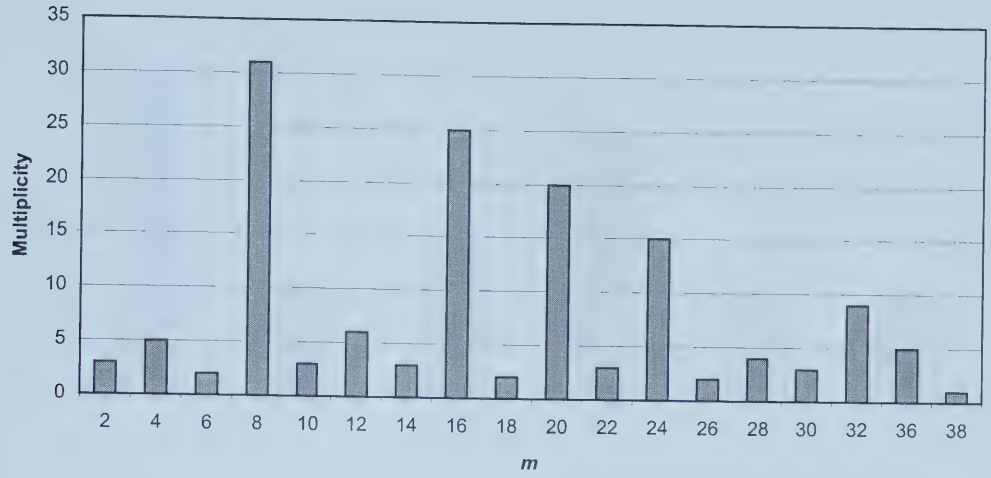


(a): Number of length-7+1 critical sequences mapped to length-7*m*+1 critical sequences.

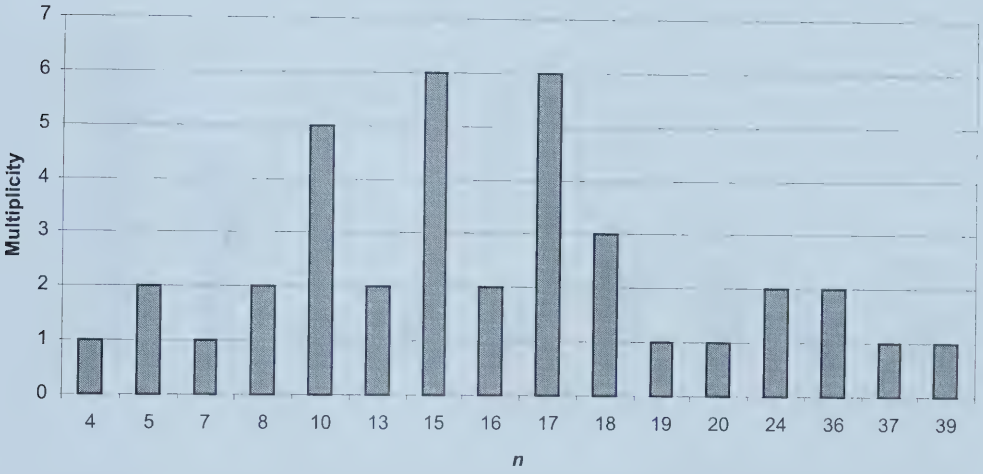


(b): Number of length-7*n*+1 critical sequences mapped to length-7+1 critical sequences.

Figure 4-7: Number of critical sequences for *K*=4 constituent codes mapped to critical sequences by the prime interleaver of size 640.

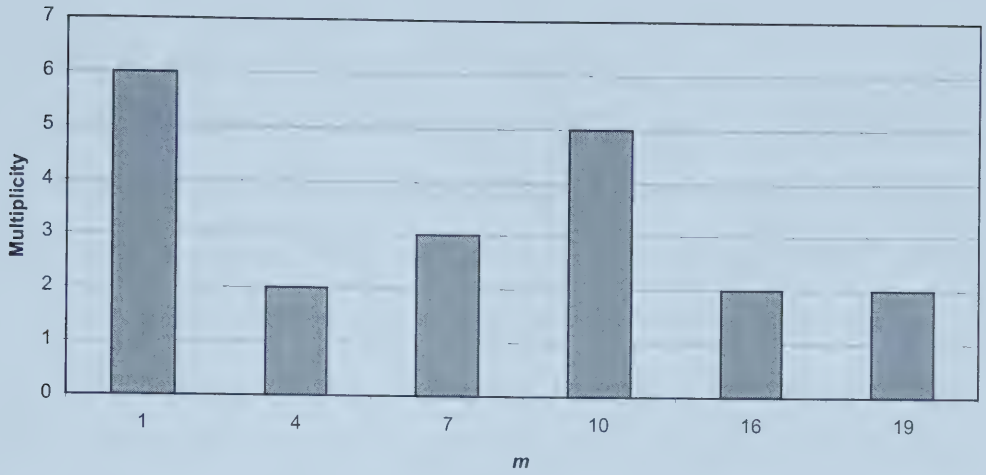


(a): Number of length-15+1 critical sequences mapped to length-15m+1 critical sequences.

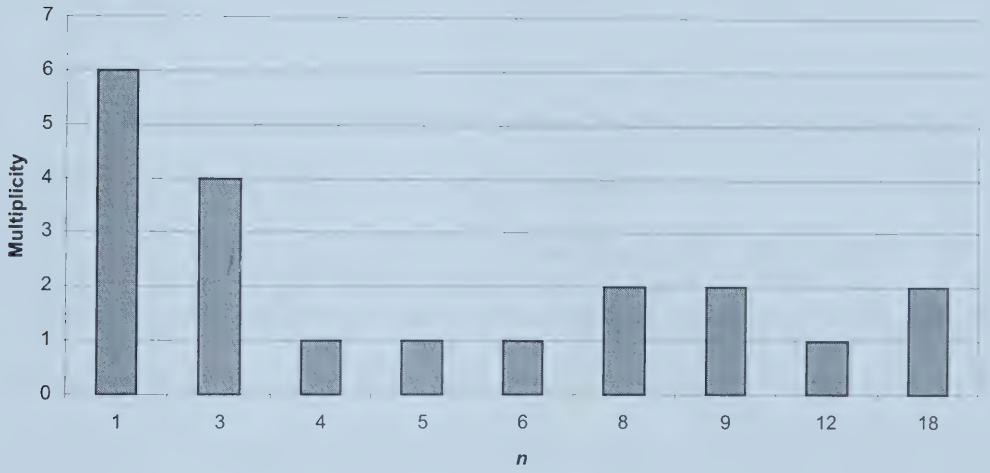


(b): Number of length-15n+1 critical sequences mapped to length-15+1 critical sequences.

Figure 4-8: Number of critical sequences for K=5 constituent codes mapped to critical sequences by the prime interleaver of size 640.



(a): Number of length-31+1 critical sequences mapped to length-31*m*+1 critical sequences.



(b): Number of length-31*n*+1 critical sequences mapped to length-31+1 critical sequences.

Figure 4-9: Number of critical sequences for *K*=6 constituent codes mapped to critical sequences by the prime interleaver of size 640.

The simulation results of Figure 4-6 can therefore be explained by the observation that the prime interleaver is better at permuting critical sequences that are less critical for K=4 constituent codes than for the high constraint length constituent codes.

4.3.2 Error event

The union bound calculations in the previous section were based on the assumption that sequences of weight-two and three dominate the performance of turbo codes. Intuitively, the average length of these sequences for codes of higher constraint length is greater than codes of lower constraint length. Naturally, there are less critical sequences (sequences of weight-two) for codes of higher constraint lengths than codes of lower constraint lengths. Performance is expected to be superior for codes of higher constraint lengths, however, results in the above section indicate otherwise.

To further investigate the effect of constraint length on the performance of turbo codes, the performance of the turbo codes considered above are analyzed in terms of their error events. An error event occurs when the decoded path through the trellis departs from the true path and then, either remerges with the true trellis or runs unmerged to the end of the trellis. The average number of data bit errors per error event and the average number of error events per errored frame provides insight into the error performance of bit error input sequences with different weights. The overall BER can then be evaluated according to :

$$\text{BER} = \frac{\text{data bit errors}}{\text{error event}} \times \frac{\text{error events}}{\text{errored frame}} \times \frac{\text{errored frames (FER)}}{\text{total frames}} \times \frac{1 \text{ frame}}{\text{total bits (size of frame)}} \quad (73)$$

Codes of shorter constraint length are known to have smaller free distances. It is assumed that shorter constraint length codes have weaker coding strength. However, this smaller distance also indicates that the average length of error events is usually

smaller compared to codes of greater constraint length. Therefore, it is generally accepted that the number of data bit errors per error event is usually lower for codes of smaller constraint length. With this assumption, a code of greater constraint length is only expected to perform better if the error events per errored frame and the FER are significantly lower than the code of smaller constraint length. This is generally true for conventional convolutional codes, therefore, this should hold true for turbo codes as well. The following sections will prove that this is not the case for turbo codes. The first three terms of equation (73) will be analyzed individually to explain the discrepancy between the expected results and simulated results.

Figure 4-10 shows the average number of errors per error event for turbo codes of constraint length 3 through 6 with an interleaver size of 640. As expected, the number of bit errors per error event is proportional to the constraint length of the codes. Also indicated by the figure is that signal-to-noise ratio has little effect on the average number of errors per error event for turbo codes.

The second contributing term of the equation is the average number of error events per errored frame. It is expected that the number of error events decreases as the signal-to-noise ratio increases. Given that the average length of error events for large constraint length codes is expected to be greater than the smaller constraint length counterpart, the larger constraint length codes have less average number of error events per errored frame. Figure 4-11 shows the combined effect of the two

contributing factors for turbo codes of constraint length 3 through 6 with an interleaver size of 640.

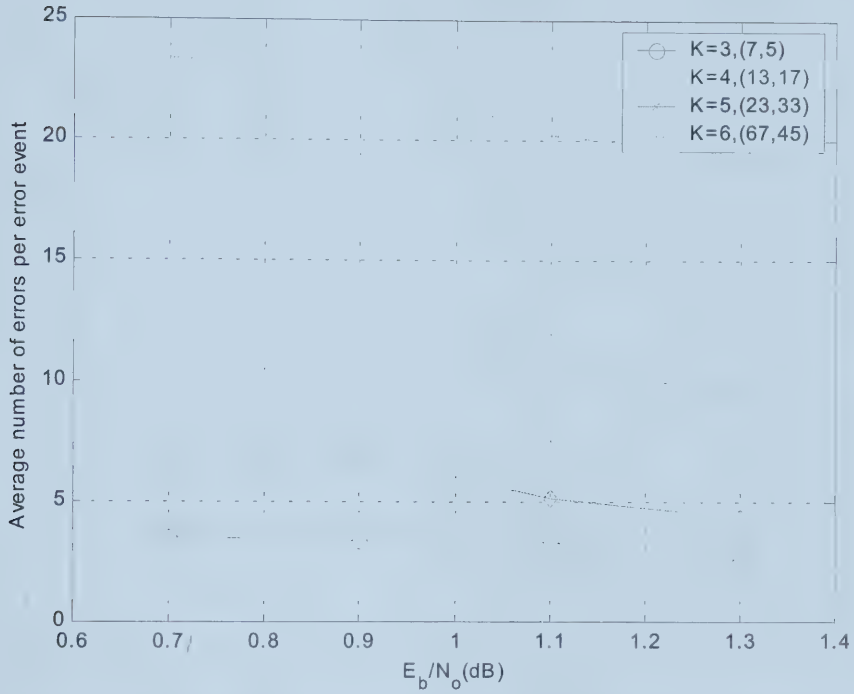


Figure 4-10: Average number of errors per error event for codes of constraint length 3, 4, 5 and 6.

Turbo codes of constraint lengths 3 through 5 follow the expected decreasing trend as signal-to-noise ratio increases, but turbo codes of constraint length of 6 do not. At high signal-to-noise ratios, codes of constraint length 3 showed the lowest average number of error events per error frame. This unexpected result may be caused by the interleaver and will be discussed later.

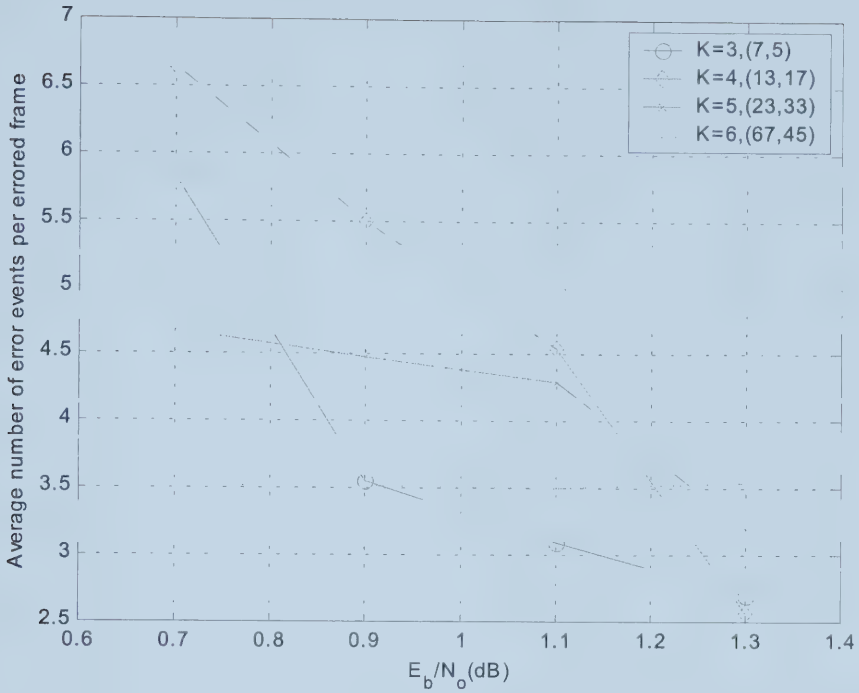


Figure 4-11: Average number of error events per errored frame.

The third factor of equation (73) is the frame error rate (FER) of the code. The FER of convolutional codes of high constraint lengths is generally significantly lower than for shorter constraint length codes. Unfortunately, in turbo codes the effect of an interleaver has to be accounted for. Figure 4-12 shows the average number of error events per transmitted frame. This is a combination of the second and the third factor of equation (73). This clearly shows the coding strength of each code considered. As expected, the code of constraint length 3 is worst. Again, the performance of turbo codes of constraint length 6 did not perform as expected. This result may be explained and analyzed in terms of the distribution of the errored bits and error events.

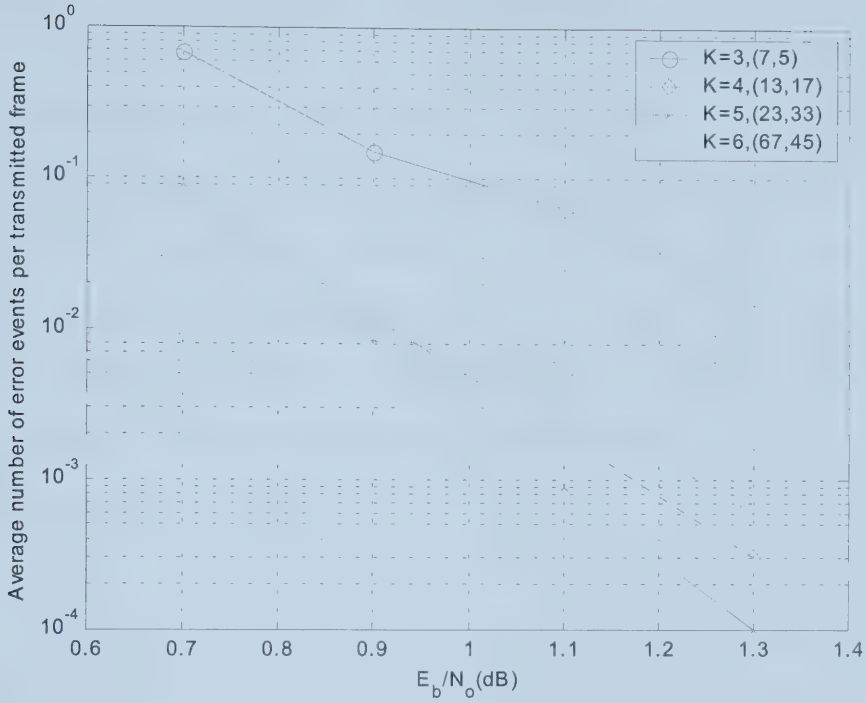


Figure 4-12: Average number of error events per transmitted frame.

Figures 4-13 through 4-16 present simulation results for the distribution of the number of data bit errors per error event for turbo codes using constituent codes with constraint lengths 3 through 6 at E_b/N_o values of 0.7 through 1.3 dB. In this simulation, the final state of the first encoder is known but the final state of the second encoder is left open. Therefore, the distribution of errors for both decoders is shown separately.

As shown in Figures 4-13 and 4-14, sequences of weight-2 and weight-3 clearly dominate the error performance at high SNR (1.3 dB) when K=3 and K=4. The interesting distribution of errors per error event lies with codes of constraint length 5

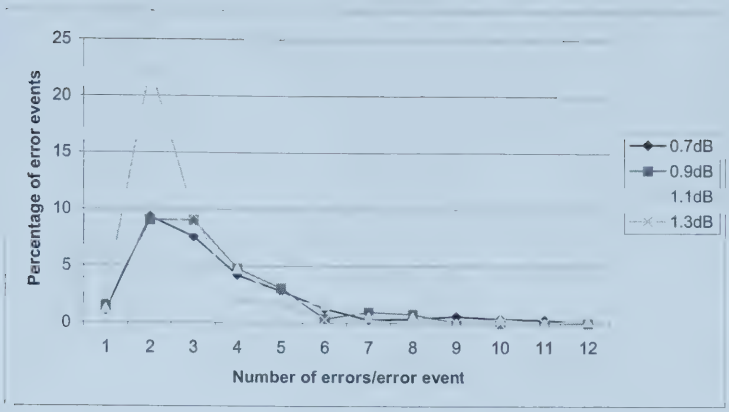
and 6. Figures 4-15 and 4-16 show that sequences of weight-two do not play as dominant a role in performance of codes with constituent code with K=5 and K=6. The distributions of errors per error event for these codes indicate that sequences of higher weight dominate. As mentioned above, the number of error bits per error event is higher for codes of higher constraint length; this simulation result should be expected. Although the average number of error events per errored frame may be lower for codes of higher constraint lengths, the average number of error bits per error event may not be able to decrease fast enough to contribute to a better error performance. This may be shown by the small performance improvement for codes of constraint lengths 5 and 6.

If the original and the interleaved sequences are well interleaved, a large number of decoded bit errors per error event usually dominates to a lesser degree as the E_b/N_o increases. This is shown by Figures 4-13 and 4-14. However, this is not the case for codes of constraint lengths 5 and 6. The occasional spikes at high decoding bit errors show the failure of the interleaver. This is seen as one of the dominating factors responsible for poor performance of the code of constraint length 6.

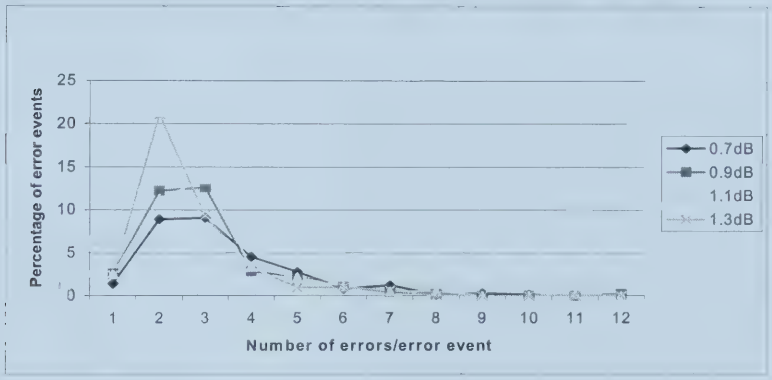
Since the average length of sequences is proportional to the constraint length of the code, the interleaver is incapable of breaking up these sequences that contribute to a high number of decoding errors per error event for both decoders, especially for a code of constraint length 6. The sequence entering both encoders/decoders is not as independent as expected. This, in turn, increases the correlation between the original

and the interleaved sequences, especially with codes of higher constraint length. This independence is critical since the decoding algorithm is iterative. With a higher number of error bits per error event and a high correlation for code of constraint length 6, its FER failed to decrease faster than for codes of constraint lengths 4 and 5, which contributed to a higher BER for a code of constraint length 6. Codes of constraint length 5 suffer from the same problem, but the effect is not as critical since the trellis for this code is usually shorter on average.

Figures 4-17 to 4-20 show the distribution of the number of error events per errored frame for different codes of interest. The E_b/N_o ranges from 0.7 dB to 1.3 dB. The coding strength of short constraint length codes is limited as shown with the high number of error events per frame. As E_b/N_o increases, the number of events per errored frame decreases. However, the distribution of the number of error events per errored frame was fairly constant for high constraint length codes. The interleaver may have caused this.

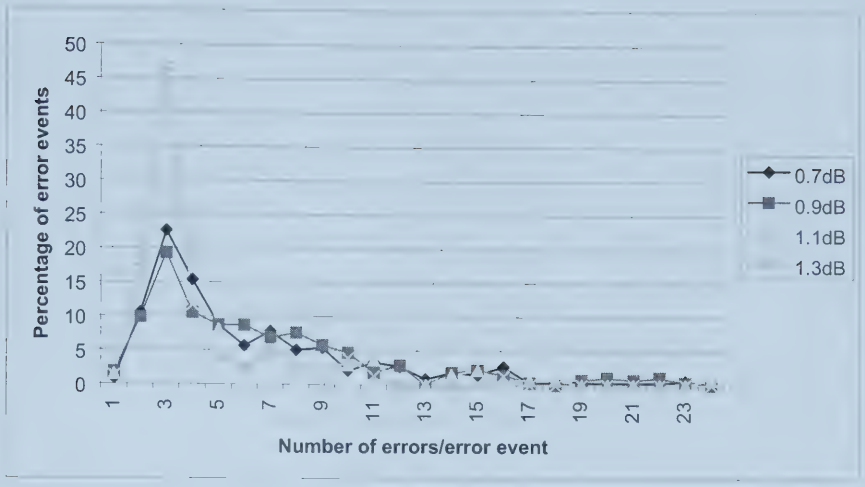


(a) Distribution of number of decoded bit errors per error event for decoder 1.

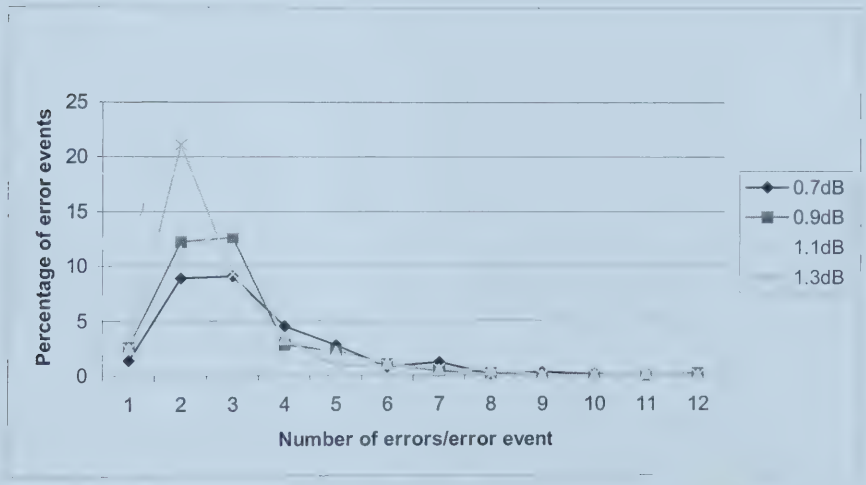


b) Distribution of number of decoded bit errors per error event for decoder 2.

Figure 4-13: Distribution of number of decoded bit errors per error event for K=3 constituent codes with PIL of size 640 with maximum 8 decoding iterations.

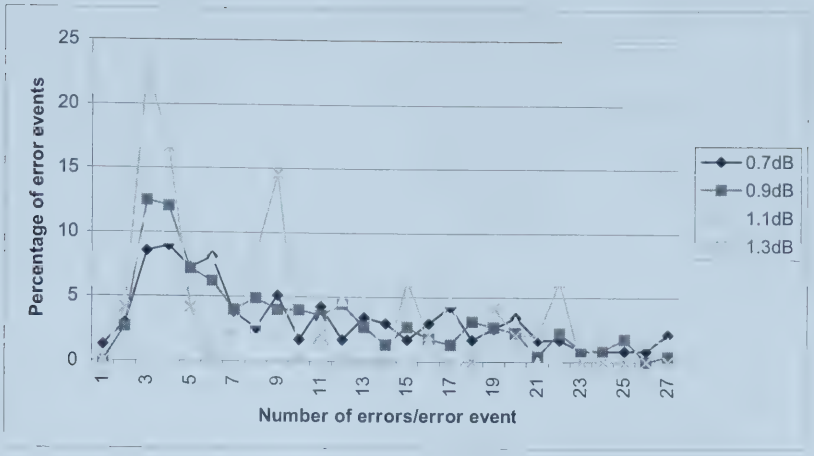


(a) Distribution of number of decoded bit errors per error event for decoder 1.

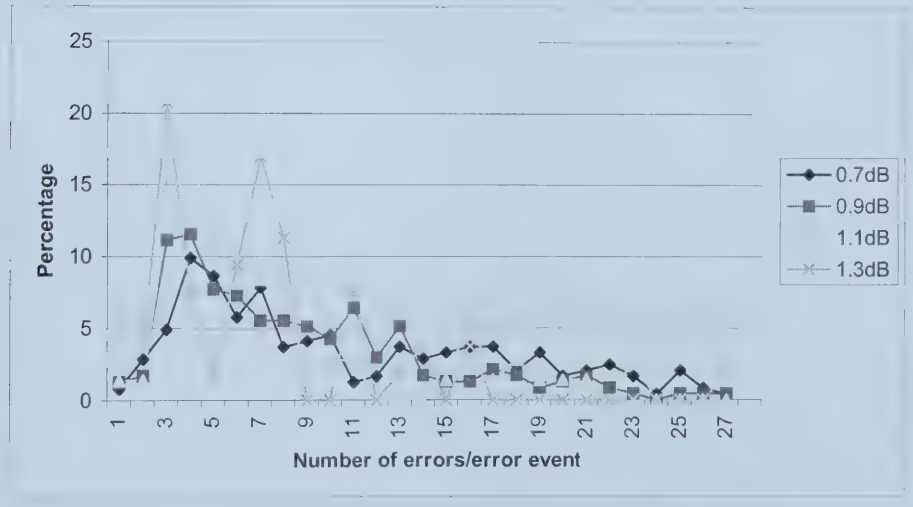


b) Distribution of number of decoded bit errors per error event for decoder 2.

Figure 4-14: Distribution of number of decoded bit errors per error event for K=4 constituent codes with PIL of size 640 with maximum 8 decoding iterations.

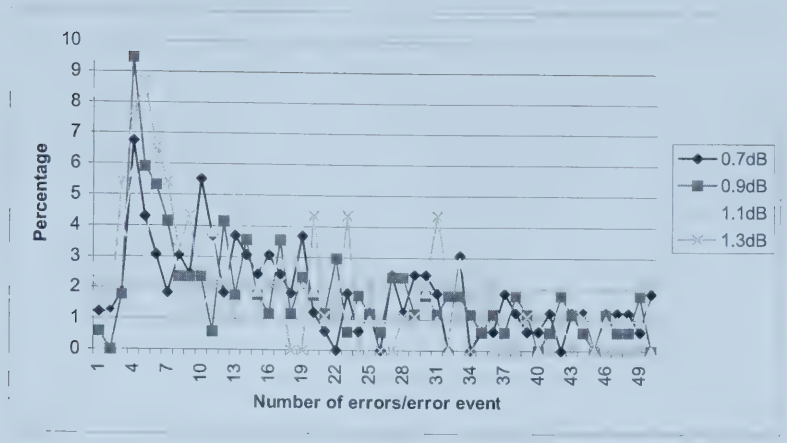


(a) Distribution of number of decoded bit errors per error event for decoder 1.

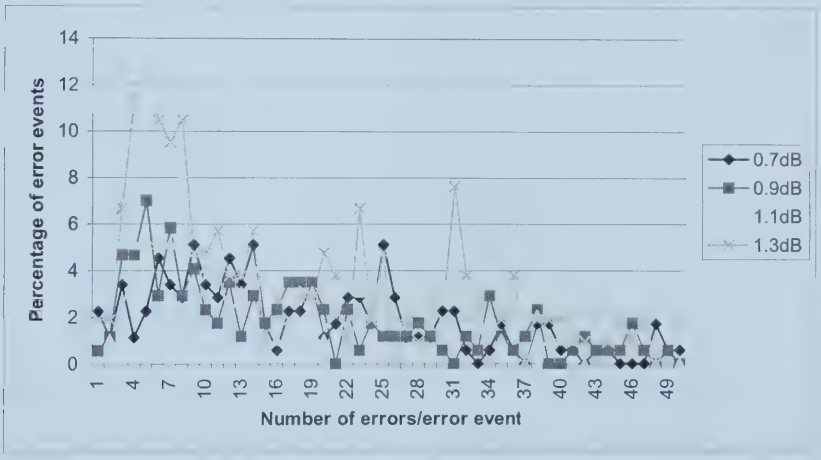


b) Distribution of number of decoded bit errors per error event for decoder 2.

Figure 4-15: Distribution of number of decoded bit errors per error event for K=5 constituent codes with PIL of size 640 with maximum 8 decoding iterations.

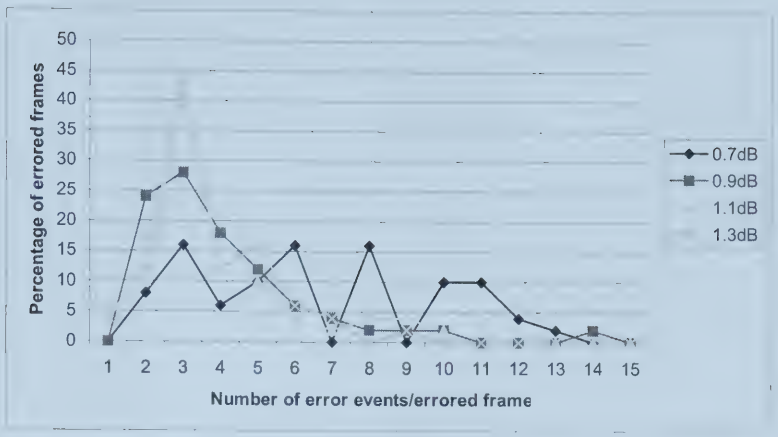


(a) Distribution of number of decoded bit errors per error event for decoder 1.



b) Distribution of number of decoded bit errors per error event for decoder 2.

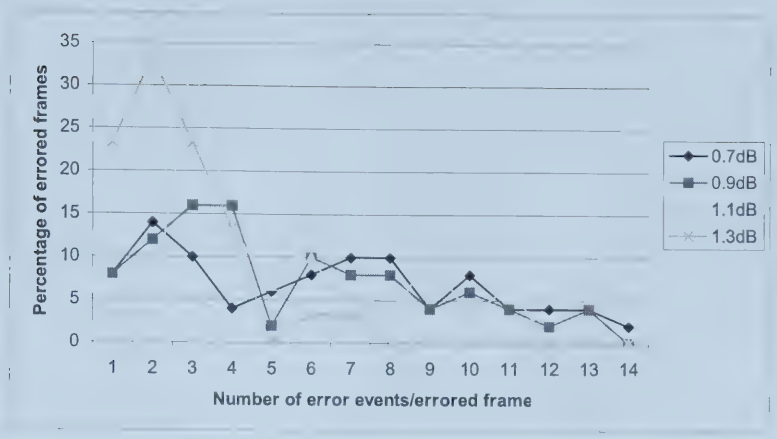
Figure 4-16: Distribution of number of decoded bit errors per error event for K=6 constituent codes with PIL of size 640 with maximum 8 decoding iterations.



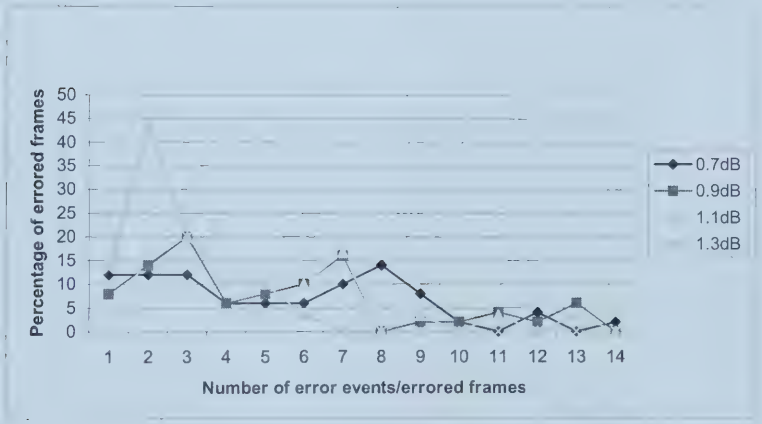
a) Distribution of number of error events per errored frame for decoder 1.

b) Distribution of number of error events per errored frame for decoder 2.

Figure 4-17: Distribution of number of error events per errored frame for K=3 constituent codes with PIL of size 640 with maximum 8 decoding iterations.

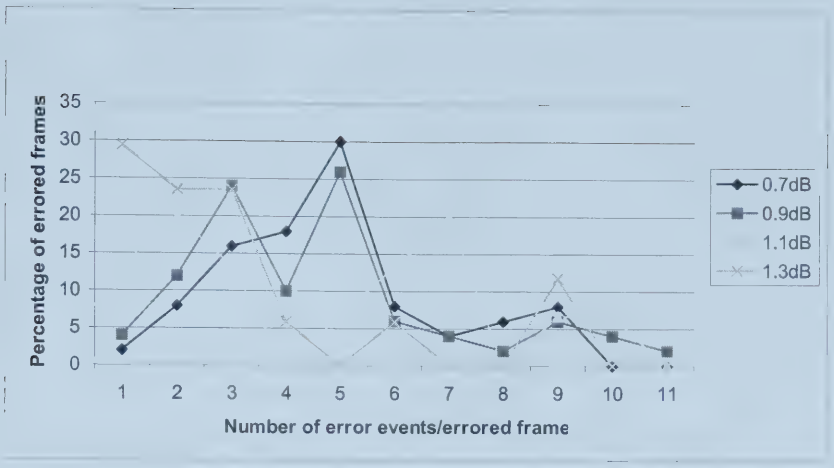


a) Distribution of number of error events per errored frame for decoder 1.

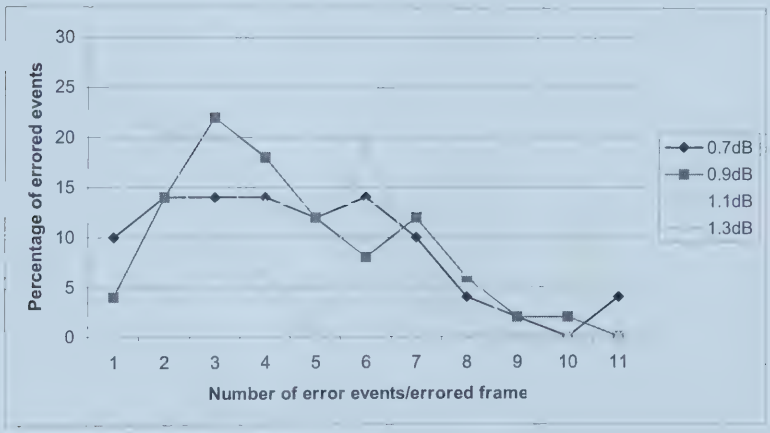


b) Distribution of number of error events per errored frame for decoder 2.

Figure 4-18: Distribution of number of error events per errored frame for K=4 constituent codes with PIL of size 640 with maximum 8 decoding iterations.

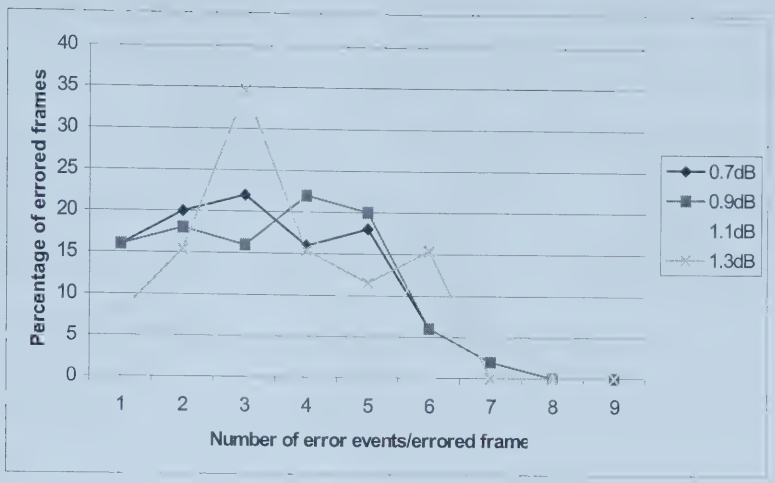


a) Distribution of number of error events per errored frame for decoder 1.

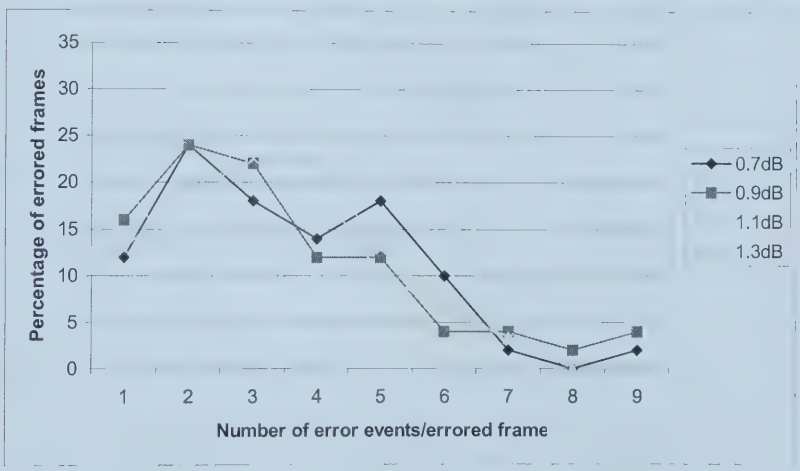


b) Distribution of number of error events per errored frame for decoder 2.

Figure 4-19: Distribution of number of error events per errored frame for K=5 constituent codes with PIL of size 640 with maximum 8 decoding iterations.



a) Distribution of number of error events per errored frame for decoder 1.



b) Distribution of number of error events per errored frame for decoder 2.

Figure 4-20: Distribution of number of error events per errored frame for K=6 constituent codes with PIL of size 640 with maximum 8 decoding iterations.

4.3.3 Constituent Code Polynomials

Figure 4-3 in the previous section showed truncated performance bounds for (13,15), (13,17), (23,35) and (23,37) constituent codes. The bound suggested that (23,37) has the best performance. Therefore, it might be expected that a turbo code comprising constituent codes of (23,37) should have the best performance among the evaluated codes. The simulation results shown in Figure 4-2, however, suggest otherwise. This adverse effect of code (23,37) is discussed in this section in terms of its distance distribution.

Table 4-1 has four columns with the first column showing the constituent codes. The second column shows the three lowest coded distances for the evaluated constituent codes where the first distance in this column will be the d_{free} of the constituent code. The third column of the table shows the number of unique sequences that would produce an encoded sequence of this distance. The last column shows the weight of these unique input sequences.

As mentioned in the Chapter II, d_{free} is an indication of the error performance of the code, and a code that has a high d_{free} will have better error performance than its lower counterpart. Also indicated by this table are the number of coded sequences of low weight that are generated by low weight input sequences. Given that indication, the distribution of these low weight sequences will give a insight into the poor performance of the (23,37) code.

Table 4-1 suggests that constituent codes (13,15) and (23,37) for low input weight are distributed at the lower distance spectrum. They are therefore expected to have poor performance among the evaluated codes. In view that (23,37) has a longer constraint length, these low distance low input weight codes are expected to be longer. Again, the interleaver may have failed to separate these sequences thus causing (23,37) to have the worst performance

Constituent Codes	Coded Sequence Weight d	Number of Sequences a_d	Input Sequence Weight
(13,15), K=4	6 8 10	2 10 49	3 2,4,6 3,5,7
(13,17), K=4	6 7 8	1 3 5	4 3 2,4,6
(23,33), K=5	7 8 9	2 2 6	3,5 4 3,5
(23,37), K=5	6 8 10	1 6 32	4 3,4,6 3,5,7

Table 4-1 Distance Spectrum of Constituent Codes as a Function of Coded Sequence Weight.

4.4 Trellis termination

As discussed in Section 2-8, there are two ways of terminating both turbo encoders. It is expected that turbo codes with dual termination will outperform turbo codes with single termination. Figures 4-21 and 4-22 show the BER and FER performance

for joint termination, dual tail termination and single termination. As seen in these figures, dual tail termination only outperforms single termination and joint termination at low E_b/N_o.

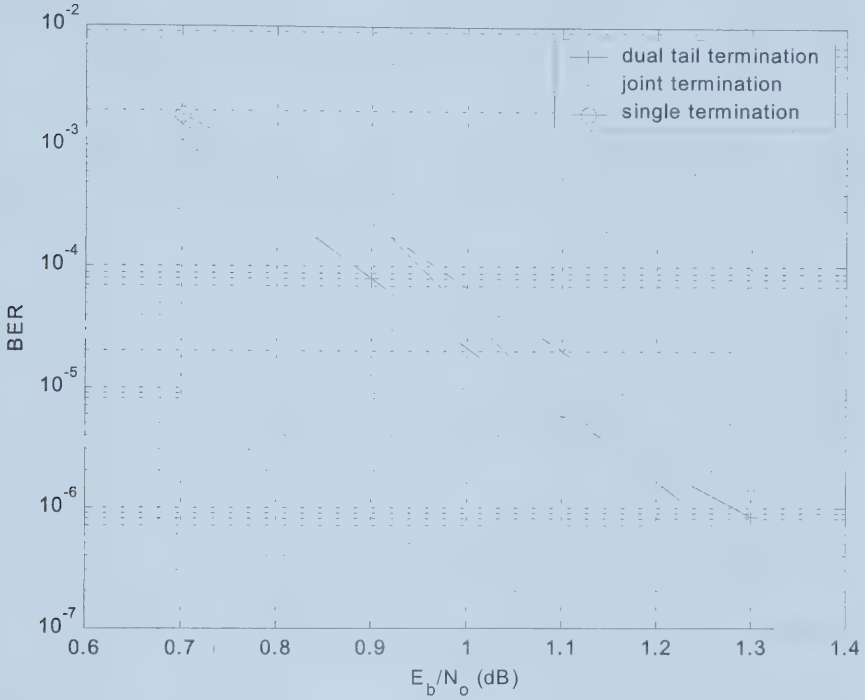


Figure 4-21: BER performance for turbo codes with different termination techniques; constraint length *K*=5, (23,33), maximum 8 iterations.

The worse performance by joint termination indicates high correlation of bit sequences. Joint termination uses the terminating bits to force both encoders to return to an all-zero state. By this method, they will be able to exchange all extrinsic information between the two encoders. Intuitively, this should be better than dual tail termination and single termination. However, the results showed no improvement for a frame size of 640, therefore, the overhead bits and the processing latency is not justified.

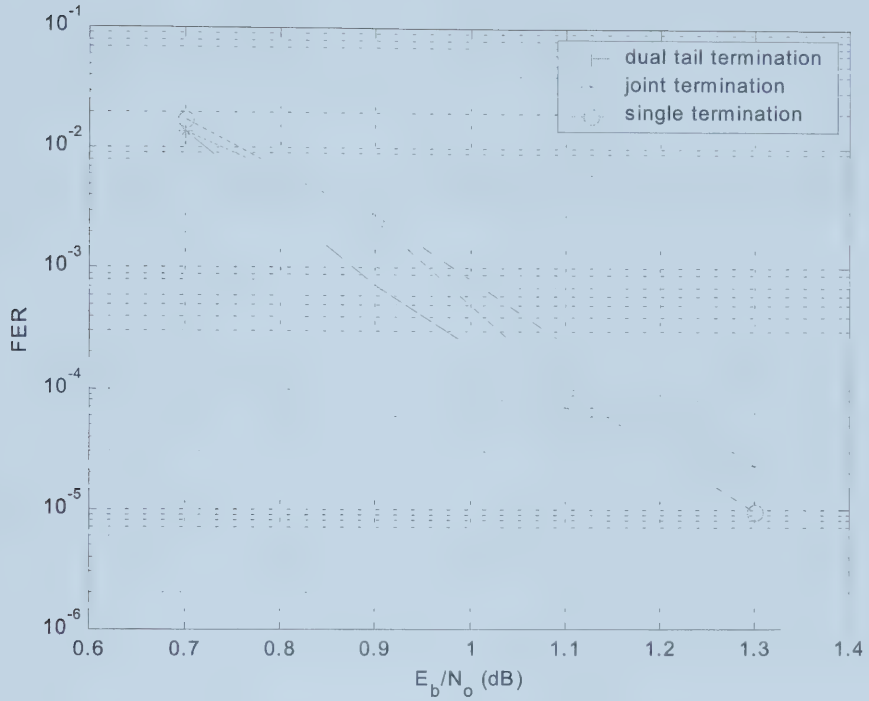


Figure 4-22: FER performance for turbo codes with different termination techniques; constraint length *K*=5, (23,33), maximum 8 iterations.

Conversely, the dual termination method terminates both encoders with different tail bits meaning that during decoding, extrinsic information regarding the tail bits cannot be exchanged between the decoders and therefore the reliability of these decoded bits will not increase with iteration. The degradation in the performance that arises from the lack of exchange of this extrinsic information is only evident at high signal-to-noise ratios in Figures 4-21 and 4-22.

The unexpected results concerning the small performance gain from terminating both encoders by individual tail bits might be explained by the limitation of the size

of the interleaver. The encoded sequence was more independent than the other two methods, thus explaining the superior performance. The terminating bits by joint termination, on the other hand, are the result of the pre-calculation of the original information sequence; the independence between the original and the interleaved sequences is lessened, increasing the correlation between the two sequences.

4.5 Conclusion

The function of the interleaver in turbo codes is to randomize the information sequence from the first encoder such that the encoded sequences that are generated by the two encoders are as independent as possible. Clearly, larger interleavers will permit increased randomness, therefore, it is not surprising to have observed an increase of at least 0.5 dB in coding gain each time the interleaver size was doubled.

Next, the ability of spread random and prime interleavers to break critical sequences was investigated and was used to explain the simulation results. Based on the assumption that low weight code sequences are the dominating factor in error performance, the union bound analysis was limited to input sequences of weights 2 and 3. The result indicated that the prime interleaver is better at breaking up these critical sequences.

Iterative decoding is a practical decoding method that allows turbo codes to achieve their superior performance. As expected, increasing the number of decoding iterations improves the performance of turbo codes, however, the degree of

improvement decreases as the size of the interleaver decreases. This was shown by the insignificant improvement with increased decoding iterations for an interleaver of size 160 as compared to an interleaver of size 640.

In this investigation, it was also shown that greater constraint length codes do not necessarily guarantee improved performance. The constituent codes used in turbo codes have to be chosen in conjunction with the interleaver. Intuitively, it might be expected that the performance of a turbo code improves with increasing constraint length of the constituent codes. Counter to expectation, it was shown in this chapter that the performance of greater constraint length codes is sensitive to the size of the interleaver. Codes of greater constraint lengths were shown to perform worse than smaller constraint length codes in many situations. As shown in the analysis, there are more unseparable critical sequences in the code as the constraint length of the code increases. Also shown was an error event evaluation for a number of turbo codes. From this evaluation, it was inferred that the interleaver failed to randomize the input sequence for longer constraint length codes. As a result, a high number of errors per error event remained even at high signal-to-noise ratios.

Constituent code polynomials of turbo codes were also investigated in this chapter. Despite the performance improvement suggested by the union bound, the (23,37) code of constraint length 5 has the worst simulated performance with both prime and spread interleavers, compared to the other code polynomials considered. Again, the interleaver may have failed to randomize the input sequence for longer

constraint length codes. The distribution of the low input weight codes of (23,37) at the low distance spectrum may also be one of the reasons for poor performance.

The last section of the chapter showed simulation results for trellis termination of turbo codes. It was found that dual termination did not prove to be a better alternative as compared to single termination. The use of the extra terminating bits was not justified for an interleaver size of 640. Dual tail termination terminate the original and interleaved information sequences with individual tail bits. The lack of a-priori information for these tail bits proved to be the downfall of this method. Joint termination used the original information sequence and permuted positions to calculate the termination bits. This method may have decreased the independence of the two sequences, which is clearly an issue for small interleavers, as shown in the results. This may be the reason for its insignificant improvement.

V. CONCLUSION

Demand for reliable data transmission in third generation wireless systems has generated interest in the high performance of turbo codes. The turbo codes' structure is well established, but the effects of their parameters are not fully understood. In this thesis, a few parameters of interest were investigated and discussed.

Basics of turbo codes were given in Chapter II. The constituent codes of turbo codes, interleaver, decoding algorithms and dual termination methods were briefly introduced.

The first section of the chapter introduced error performance related properties of convolutional codes. Distance and constraint length were the two most important properties for these codes. The next section discussed the selection of an interleaver in turbo codes. An algorithmic interleaver, known as the prime interleaver, and a randomly generated spread random interleaver were introduced. These interleavers were investigated in Chapter IV.

Following that section was the discussion of decoding algorithms for convolutional codes. The Max Log-Map algorithm with correction term was shown to have equivalent decode performance to the Log-Map algorithm but with less computation. This algorithm was used as the decoding algorithm in this thesis.

The last section of the chapter introduced two methods of dual termination. Dual tail termination used two individual tail bit sequences to terminate the encoded trellis. Joint termination terminated both encoded trellises by pre-computing the information sequence. The effects of these methods were investigated in Chapter IV.

The system environment for simulating turbo codes was established in Chapter III. The Inverse Fast Fourier transform (IFFT) method for generating Rayleigh random variables was used as the flat fading channel model. Features such as antenna diversity, rake receivers and power control were added to simulate a DS-CDMA system environment. This resulted in a log-normal distributed channel model with a standard deviation of 2 dB.

Chapter IV of the thesis showed simulation results for the parameters of interest of turbo codes. The results can be summarized as follow:

a) A larger interleaver provides a more random environment, which translates to better performance.
b) A good interleaver has to provide the required randomness such that encoded sequences are not correlated and it has to be capable of breaking up critical sequences.
c) Larger constraint length codes do not necessarily mean better performance in turbo codes. It was found that larger constraint length codes translate to longer sequences of low weight. With an interleaver size of 640, these sequences failed to be separated or randomized which translates into poor performance.

d) There exists a maximum decoding iteration beyond which the performance does not improve and can deteriorate.

e) Joint termination did not prove to be a better option compared to the single termination method for small interleavers.

In conclusion, this investigation has shown that it is important to select code polynomials in conjunction with the interleaver. For wireless applications, only a small interleaver is considered. As a result, the selection of the code polynomials and the method of termination will have to consider the characteristics of the interleaver and not be chosen in isolation from one another. Further investigation may include applications of the alternative turbo codes or serially concatenated codes.

References:

[1] D.N. Knisely, Q. Li and N.S. Ramesh, "CDMA2000: a third generation radio transmission technology," *Bell Labs Technical Journal*, July-September 1998, pp. 63-78.

[2] J.G. Proakis, *Digital Communications*, 3rd Edition, New York: Mcgraw Hill, 1995.

[3] G.D. Forney, Jr., *Concatenated Codes* Cambridge, MA:MIT Press 1966.

[4] C.Berrou and A. Glavieux, "Near optimum error correcting coding and decoding: Turbo codes," *IEEE Transactions on Communications*, vol. 44, no. 10, pp. 1261-1271, Oct.96.

[5] C.E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379-423 and 623-656, July and October, 1948.

[6] "Multiplexing and channel coding (FDD)," 3rd Generation Partnership Project; Technical Specification Group Radio Access Network; 3G TS 25.212 V3.1.1, Dec. 1999.

[7] Gordon L. Stuber, *Principles of Mobile Communication*, Massachusetts: Kluwer Academic Publishers, 1996.

[8] J. Hokfelt, O. Edfors, T. Maseng, "Turbo codes: Correlated extrinsic information and its impact on iterative decoding performance," *Proc. of VTC '99-Spring,* Houston, USA, May 1999.

[9] S. Benedetto, R. Garello, G. Montosi, "A search for good convolutional codes to be used in construction of turbo codes," *IEEE Transactions on Communications*, vol. 46, no. 9, pp. 1101-1105, September 1998.

[10] G. Battail, " A conceptual framework for understanding turbo codes," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 2, pp. 245-254, Feb. 1998.

[11] D. Divsalar and R.J. McEliece, "Effective free distance of turbo codes," *Electronics Letters*, vol. 32, no. 5, pp. 445-446, February 29,1996.

[12] D. Divsalra and F. Pollara, "On the Design of turbo codes," TDA Progress report 42-123, Jet Propulsion Laboratory, Pasadena, California, November 15, 1995, pp. 99-121.

[13] D. Divsalar and F. Pollara "Multiple turbo codes," *Proc. of MILCOM* 95, pp. 279-285.

[14] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate." *IEEE Transactions on Information Theory*, col. IT-20, pp. 284-287, March 1974.

[15] C. Heegard and S. Wicker, *Turbo Coding*, Massachusetts: Kluwer Academic Press, 1999.

[16] J.Hagenauer and P. Hoeher, "A Viterbi Algorithm with soft decision outputs and its applications," in *Proc. GLOBECOM '89*, November 1989, pp. 1680-1686.

[17] P. Robertson, E.Villebrun, P. Hoeher, "A comparison of optimal and sub-optimal MAP decoding algorithms operating in the log domain," *Proc of ICC'95*, pp. 1009-1013.

[18] J.A. Erfanian, S. Pasupathy and G. Gulak, "Reduced complexity symbol detectors with parallel structures for ISI channels," *IEEE Transactions on Communications*, vol. 42, pp. 1661-1671, February/March/April 1994.

[19] D. Divsalar and F. Pollara. "Turbo codes for PCS application," *Proc of ICC 95*, pp. 54-59.

[20] A. Shibutani, H. Suda , F. Adachi, "Complexity reduction of turbo decoding," *Proc. of VTC 99'* –Fall, Amsterdam, The Netherlands, September 1999, vol. 3, pp. 1570-1574.

[21] F. Zhai and I. J. Fair, "New error detection techniques and stopping criteria for turbo decoding," *Proc. CCECE 2000*, pp. 58-92.

[22] P. Robertson, "Illuminating the structure of decoders for parallel concatenated recursive systematic (turbo) codes," *Proc. IEEE Globecom Conf.*, 1994, pp. 1298-1303.

[23] P. Guinand, J. Lodge, "Trellis termination for turbo codes," *Proc. 17th Biennial Symposium on Communications*, Kingston ON, Canada, May 1994, pp. 389-392.

[24] H. Mizuguchi, A. Aoyama, S. Yoshida and A. Usirokawa, "Performance Evaluation on power control and diversity of next generation CDMA system," *IEICE Transactions on Communications*, vol. E81-B, no. 7, pp. 1345-1353, 1998.

[25] W.C. Jakes, *Microwave Mobile Communcations,* IEEE Edition, New York: IEEE press, 1994.

[26] M.F. Pop, N.C. Beaulieu, "Limitations of sum of sinusoids fading channel simulators," *IEEE Transactions on Communications,* vol. 49, pp. 699-708, April 2001.

[27] D.J. Young, N.C. Beaulieu, "The Generation of Correlated Rayleigh Random Variates by Inverse Discrete Fourier Transform," *IEEE Transactions on Communications,* vol. 48, pp. 1114-1127, July 2000.

[28] S. Ariyavisitakul and L.F. Chang, "Signal and interference statistics of CDMA system with feedback power control," *IEEE Globecom 1991*, Phoenix, AZ. pp. 1490, Dec.1991.

[29] TIA/TR45.5, The cdma2000 ITU-R RTT Candidate Submission, July 1998.

[30] T. Dohi , Y. Okumura, F. Adachi, "Further Results on Field Experiments of Coherent Wideband DS-CDMA mobile radio," *IEICE Transactions on Communications*, vol. E81-B, no. 6, pp. 1239-1246, 1998.

[31] P.W. Baier, P. Jung and A. Klein, "Taking the challenge of multiple access for third generation cellular mobile radio systems–a European view," *IEEE Communications Magazine*, pp. 82-89, February 1996.

[32] F. Swarts, P van Rooyan, I. Oppermann and M.P.Lotter, *CDMA techniques for third generation mobile system*, Massachusetts: Kluwer Academic Publishers, 1999.

[33] J. Rothweiler, "Turbo codes," *IEEE Potentials*, vol. 18, no. 1, pp. 23-25, Feb/Mar 99.

[34] B. Sklar, *Digital Communications*, New Jersey: Prentice Hall, 1988.